HANDSHAKING PROTOCOLS AND JAMMING MECHANISMS FOR
BLIND RENDEZVOUS IN A DYNAMIC SPECTRUM ACCESS
ENVIRONMENT

THESIS

Aaron A. Gross, 2nd Lieutenant, USAF

AFIT/GCO/ENG/10-09

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

HANDSHAKING PROTOCOLS AND JAMMING MECHANISMS FOR
BLIND RENDEZVOUS IN A DYNAMIC SPECTRUM ACCESS
ENVIRONMENT

THESIS

Aaron A. Gross, 2nd Lieutenant, USAF

AFIT/GCO/ENG/10-09

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

AFIT/GCO/ENG/10-09

HANDSHAKING PROTOCOLS AND JAMMING MECHANISMS FOR BLIND

RENDEZVOUS IN A DYNAMIC SPECTRUM ACCESS ENVIRONMENT

THESIS

Presented to the Faculty

Department of Electrical and Computer Engineering

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

In Partial Fulfillment of the Requirements for the

Degree of Master of Science

Aaron A. Gross
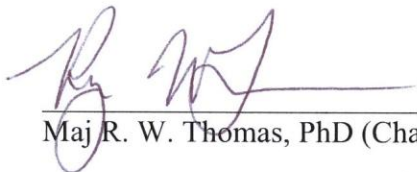
2nd Lieutenant, USAF

March 2010

AFIT/GCO/ENG/10-09

HANDSHAKING PROTOCOLS AND JAMMING MECHANISMS FOR BLIND

RENDEZVOUS IN A DYNAMIC SPECTRUM ACCESS ENVIRONMENT

Aaron A. Gross
2nd Lieutenant, USAF

Approved:

Maj R. W. Thomas, PhD (Chairman)   16MAR10
                                                Date

Maj M. D. Silvius, PhD (Member)   16 Mar 2010
                                                Date

Dr. R. K. Martin (Member)   16 Mar 2010
                                                Date

Abstract

Blind frequency rendezvous is an important process for bootstrapping communications between two radios without the use of pre-existing infrastructure or common control channel. Blind rendezvous is especially useful in Dynamic Spectrum Access (DSA) networks where frequency holes frequently change due to primary user activity. In a blind rendezvous process, two radios attempt to arrive at the same channel in a potentially dynamic spectral environment (without pre-coordination) and recognize the presence of each other.

This thesis has two aims: First, it refines existing blind rendezvous techniques by introducing a handshaking algorithm for setting up communications once two radios have arrived in the same frequency channel. Second, it investigates the effect of different jamming techniques on blind rendezvous algorithms that utilize this handshake. Mathematical analysis is used to determine the probability of a successful handshake, as well as the expected time to complete the handshake. We investigate the handshake performance in conjunction with various rendezvous algorithms from literature, .determining the difference between the algorithm's time to rendezvous (TTR) and time to meet (TTM) (the time required to arrive in the same channel but not necessarily rendezvous).

The presence of jammers can affect both the blind rendezvous and handshaking algorithms. Four different jamming techniques are applied to the blind rendezvous process: noise, deceptive, sense, and primary user emulation (PUE). Each jammer type is

analyzed to determine how they increase the TTR, the probability of jamming, and how long it takes to jam.

We use a combination of mathematical analysis and simulation to measure the performance of the handshake and the jammers. The handshake increased the TTR from approximately 40% to 360%, depending on the amount of time spent beaconing and receiving. As expected, each of the jamming techniques increased the TTR. The noise and deceptive jammers were the least effective, increasing TTR by at most 8%. The sense jammer was the most effective, increasing TTR by up to 45%. The PUE jammer was in the middle with increases of up to 25%. Furthermore, all jammers were more effective in increasing the TTR against the modular clock algorithm than against the random algorithm.

Acknowledgements

I would like to express my thanks to my research advisor, Maj Ryan Thomas, for his exceptional guidance and support throughout the course of this thesis effort.

Aaron A. Gross

Table of Contents

List of Figures

List of Tables

HANDSHAKING PROTOCOLS AND JAMMING MECHANISMS FOR BLIND

RENDEZVOUS IN A DYNAMIC SPECTRUM ACCESS ENVIRONMENT

I. Introduction

Wireless communications have become a staple of modern society, promoted by newer

technologies such as netbooks and smartphones. The boom of wireless technology has

also put a strain on the available spectrum and resources for data transaction. Being able

to efficiently utilize available spectrum and resources becomes increasingly attractive.

Using smart, spectrum adaptable devices can help solve this problem.

Furthermore, it is important for certain applications that these radios be able to

operate in hostile or disaster environments in which communications infrastructure may

be damaged, destroyed, or otherwise unavailable. Under this environment, the spectrum

may be crowded by non-compliant devices, and the infrastructure that spectrum users

depend on may not be present.

The military faces similar problems. As the current spectrum is delegated to more

commercial purposes, other users --such as the Air Force-- are left with fewer options for

leveraging the available spectrum. Due to this emerging problem, the Air Force

Scientific Advisory Board (SAB) conducted a study in 2008 to determine the

"Implications of Spectrum Management for the Air Force" [9]. The study emphasized

the need for "spectrum mutability," which is the concept that a system can mutate its

spectrum use in order to adapt to changes in the environment.

Dynamic Spectrum Access (DSA) is an emerging technology that can help alleviate these needs for civilian society, hostile environments, and the military. DSA attempts to have systems use under-utilized areas of the frequency spectrum without interfering with primary users. In its most extreme implementations, DSA does not even require any infrastructure for deployment. Current DSA implementations require cognitive systems (such as cognitive networks and cognitive radios [27]) that are capable of adapting to changing spectrum.

1.1 Research Problem and Scope

A problem arises when a cognitive system operating in a DSA environment needs to establish communications. This requires that the cognitive radios find each other first. The process with which two radios attempt to discover each other is known as rendezvous. The rendezvous subproblem that occurs when there is no pre-existing communications infrastructure is called blind rendezvous.

When attempting to rendezvous, it is important that radios can quickly find each other as spectrum availability changes over time. In infrastructureless and hostile environments, it is even more imperative that this rendezvous process can be achieved quickly.

Much rendezvous research operates under the assumption that rendezvous is completed once two radios are in the same frequency channel. However, similar to other communication models, the two radios must have a method for establishing communications once they are in the same channel. This is a related problem to neighbor discovery and the hidden node problem of wireless networks, but made more difficult by the fact that there is a large, dynamically changing frequency space to search through.

As wireless networks expand in size and popularity, so also do the threats against their integrity. One serious threat is radio interference and jamming. Introducing a method for communications setup can bring along new vulnerabilities to the rendezvous process.

This thesis aims to develop a handshaking algorithm that two radios can use when in the same spectrum band. The handshake is analyzed to determine the probability of success and the expected time to complete. Further, its performance is measured in the context of rendezvous algorithms by determining the increase in the time to rendezvous (TTR) with the handshake, compared to the time to just meet in the same channel.

The thesis then analyzes the effects of different jamming techniques against this handshake. The jammers are analyzed to determine how they increase in TTR, the probability of jamming, and how long it takes to jam.

All analysis in the thesis stops at the point where rendezvous occurs, meaning we do not analyze any effects *after* rendezvous has occurred.

1.2 Approach

To develop a practical handshaking algorithm, we investigate other forms of handshaking such as neighbor discovery and MAC protocols. Once the handshaking algorithm is defined, we simulate it under both random and modular clock rendezvous algorithms [26] to determine its performance. The objective is to define and characterize a general protocol for meeting once two radios are in the same channel.

After the handshake has been analyzed, we introduce a malicious radio into the rendezvous process. This malicious radio utilizes various jamming techniques in order to disrupt the handshake and prevent rendezvous. The various techniques used are

simulated and analyzed in their effectiveness against both rendezvous algorithms. The objective is to show how effective these jamming techniques are at increasing the time required to rendezvous.

In Chapter 2, we investigate several foundational technologies for rendezvous and cognitive radios, rendezvous algorithms, handshaking methods, and various jamming strategies. In Chapter 3, we model the handshake algorithm and further define the jamming techniques. In Chapter 4, we perform a mathematical analysis of the handshake algorithm, and showcase the simulation results of both the handshake and jamming techniques. Finally, Chapter 5 will detail the conclusions drawn from the simulation and analysis.

II. Background

This chapter will discuss some of the foundational technologies in the rendezvous and cognitive radio domain. The chapter will then examine what rendezvous is and various models used to solve the rendezvous problem. Next, existing handshaking methods are investigated with respect to their application to this problem. Finally, jamming ideologies and methods are discussed.

2.1 Foundational Technologies

While the concept of rendezvous has been around for a while, recent technologies have allowed for real world application of these theories. Not only have some of these technologies enabled the use of rendezvous, but have also shown the need for an efficient means by which to do so.

2.1.1 Software Defined Radios

A Software Defined Radio (SDR) is defined by IEEE as a '*radio in which some or all of the physical layer functions are software defined.*' This software implementation allows SDRs to be flexible and customizable towards a variety of applications. Some of the important physical layer properties that have been translated into software are the carrier frequency, signal bandwidth, modulation, network access, and in some cases cryptography and data encoding.' The flexibility in SDRs has led to the implementation of features that allow the SDR to optimize its performance. These features allow for capabilities that will turn a SDR into a cognitive radio (CR) [12]

2.1.2 Cognitive Radios

First presented by Mitola in [19] the capabilities present on SDRs allow features to be implemented in order to allow the radio to make decisions and optimize

performance, such a radio is also known as a cognitive radios (CRs). The SDR Forum and IEEE defines CRs as

> '*radios in which communications systems are aware of their environment and internal state, and can make decisions about their radio operating behavior based on that information and predefined objective* [12] .'

The decisions made by the CRs allow them to support three entities: the user, the spectrum regulator, and the network operator. Users are important to CRs, specifically the user objectives. If users want fast response times, the CRs can analyze the current network conditions and attempt to optimize performance to better meet the user objectives. The spectrum regulator is in charge of allocating spectrum to different users. With the numerous and ever increasing amount of users in the global telecommunications market, finding and allocating spectrum is difficult. CRs adaptability can allow them to easily switch spectrums as necessary to allow for other users to fit in. This adaptability also supports the Dynamic Spectrum Access (DSA) movement that attempts to allocate under-utilized parts of the spectrum.

2.1.3 Dynamic Spectrum Access

In the United States and other countries, most of the wireless frequency spectrum has been allocated to different users (such as cellular providers, television stations, radars, etc). Despite this static allocation, many of these frequency ranges are under-utilized by their owner. However, because these ranges are licensed to specific primary users (PUs), it is difficult for secondary users (SUs) to use this under-utilized spectrum. Dynamic Spectrum Access (DSA) attempts to allow SUs to use the under-utilized spectrum space without interfering with the PUs. This can be done in different ways such as having the SUs operate at lower power levels to stay under the noise envelope or by

6

hopping between different frequencies as PUs arrive [12]. Because CRs can learn from and analyze the environment, they can be used as effective DSA SUs by regulating power levels or jumping between available spectrum, minimizing interference and maximizing the utilization of available spectrum.

Rendezvous in a DSA environment is unique in that the available channels for radios to rendezvous in are constantly changing. Even once rendezvoused, radios may get "booted" from the channel and have to start over again. Therefore, the radios will have to constantly scan the frequency spectrum to determine which channels are currently available for rendezvous.

Akyildiz refers to DSA and CR networks as the NeXt generation of wireless networks [1] He mentions fours ways in which DSA/CRs can more efficiently utilize spectrum:

- Spectrum sensing, detecting unused spectrum and locations of PUs;

- Spectrum management, selecting the best available spectrum;

- Spectrum sharing, coordinating spectrum access with other SUs; and

- Spectrum mobility, switching channels when a PU arrives.

This work focuses on determining the performance of aspects of *spectrum sharing* in the presence of malicious jammers.

2.2 Rendezvous

The rendezvous problem is often described as a game in which two players are attempting to find each other in a search space. These games evolved from general search games, in which a player attempts to find either an object or an evading player.

The key change is that in rendezvous search, it is a cooperative search between the two players.

While the general rendezvous game can be applied to any generic space, such as people trying to find each other in the mall, not much research has been applied to rendezvous in the frequency domain and in wireless communications. With frequency rendezvous, two radios attempt to find each other in the same frequency channel so that they can communicate with each other. DaSilva [11] notes that frequency rendezvous is especially important in the DSA environment since the secondary users (SUs) change channels frequently, mostly due to primary users (PUs) arriving in the channel.

There are generally considered to be two types of rendezvous games: asymmetric and symmetric [3] Asymmetric rendezvous is when the two players use different strategies. Bluetooth is a specific example of asymmetric rendezvous in that different Bluetooth devices act as master or slave during rendezvous. In other rendezvous literature, some prior arrangement of strategy is made before the rendezvous process begins [13] For example, if two people are at the mall, they may agree that if they are separated, one will wait in place while the other searches. However, the implication that players can meet beforehand to discuss how to divide the strategy is not applicable to all scenarios. In these cases, where both radios are forced to use the same strategy, symmetric rendezvous is used. Since the two players will either not meet right before attempting to rendezvous or know when rendezvous may be required, they cannot use the previous strategies used in asymmetric rendezvous. Using the earlier example, assume two people are at the mall but already separated. If they try to use the same strategy as before (one waits while the other searches), they may both decide to wait and thus never

meet. Instead, they could each choose to wait for a time and search for a time, each with some probability. By using the same predefined strategy, the players could guarantee meeting, just not as quickly. While both rendezvous processes require predefined strategies, asymmetric rendezvous assumes these strategies can be updated based on knowledge the players have of each other. Symmetric rendezvous assumes that this knowledge will not always be available. In a sense, symmetric rendezvous handles the worst case scenario.

Similar to the symmetric/asymmetric differences, rendezvous is often characterized by being either synchronous or asynchronous [17] . Synchronous rendezvous is when the two rendezvousing radios are synchronized with each other in time; much like the symmetric rendezvous synchronizes their strategies. Many rendezvous algorithms assume that radios are synchronized; the radios share a common clock, enter channels at the same time, and have aligned time slots for transmitting and receiving messages. In reality, this assumption of synchronization is difficult to achieve. Asynchronous rendezvous does away with this assumption by allowing the radios' clocks to be offset relative to each other.

Sometimes radios have the benefit of a control channel to tell them which channels are available to rendezvous in. However, in infrastructureless environments, the assumption of having a control channel is not always applicable. In a DSA environment, having a dedicated control channel is unrealistic as the available spectrum is constantly changing. Also, control channels can become bottlenecks and significant points of failure. Should the available infrastructure become damaged or corrupted over time, a different method of rendezvous would be required. Attempting to rendezvous without a

control channel can be referred to as blind rendezvous. Several algorithms have been

developed in order to solve the blind rendezvous problem; three of them are random,

non-orthogonal sequence based, and modular clock.

2.2.1 Random

The simplest of these is random rendezvous in which radios randomly choose

from the available channels in an attempt to find each other. Random rendezvous often

serves as the baseline for evaluating more advanced algorithms. For a small number of

channels, especially two, random has been shown to perform well. Both [13] and [28]

involve creating optimal random strategies. Anderson and Weber [28] in particular,

combine randomly switching channels and not moving, each with some probability.

Algorithm 1 details a simple random rendezvous strategy for two radios described in [26]

Here, $c_{i,ji}$ is the random channel selected by radio i.

Algorithm 1 Random Rendezvous [26]
1: Observe m_i, the number of channels available to radio i
2: **while** not rendezvous **do**
3:     $j_i = \text{rand}[0, m_i)$
4:     $c = c_{i,ji}$
5:     attempt rendezvous on channel c
6: **end while**

2.2.2 Modular Clock Algorithm

Using a discrete math relationship usually associated with cryptography,

specifically the Chinese Remainder Theorem (CRT), Theis [25] proposes the modular

clock (MC) algorithm, which guarantees rendezvous in $O(m)$ time in many cases. To

use cryptographic number theory, each radio in the MC algorithm selects the next prime

number, p, larger than the number of observed available channels, m. It will then choose

a random starting channel, c, and a random rate, r, from between 1 and $p - 1$. Looking

at the available channels in order, the rate is how many channels in the channel sequence the radio will jump ahead each turn. To keep the radio within the bounds of the available channels, the channel entered after the jump is modulated with the prime number, p. However, this also means that the radio could end up in a channel between m and p. In this case, c is set to a random channel between 0 and $m - 1$. A timeout is also included because in some special cases, rendezvous may not occur. Algorithm 2 below details the MC process for a particular radio.

Algorithm 2 Modular Clock Algorithm (MC) [25]

1:   observe m_i, the number of channels available
2:   calculate p_i, the next largest prime to m_i
3:   $j_i^0 = rand[0, m_i)$
4:   **while** not rendezvous **do**
5:      choose r_i from $[0, p_i)$ randomly
6:      **for** $t = 0$ to $2p$ **do**
7:         $j_i^{t+1} = (j_i^t + r_i) \, mod(p_i)$
8:         **if** $j < m$ **then**
9:            $c = c_{i,j_i^{t+1}}$
10:        **else**
11:           $c = c_{i,(j_i^{t+1} \, mod(m_i))}$
12:        **end if**
13:        attempt rendezvous on channel c
14:      **end for**
15: **end while**

In Algorithm 2, r_i is the rate of channel hopping, t is the time slot of the system, p_i is the prime selected by radio i, and c_{i,j_i} is the currently selected channel for radio i, based on its randomly selected index, j_i, of its available channel list, m_i.

Since each run through the algorithm has a $1/p$ probability of choosing the same r value again, Theis shows that the probability of failing to rendezvous after p changes of r, is $1/p^p$. This is the special case in which rendezvous will not occur. Using an infinite

series, an upper bound is discovered for the expected time to rendezvous (TTR) of slightly larger than $2p$, which is asymptotical to $O(m)$ time.

Further results are derived when the MC algorithm is used for radios who do not share the same available channels. Assuming that two radios choose two distinct prime numbers, p_1 and p_2, rendezvous will occur within $p_1 * p_2$ time steps. Using the CRT, Theis shows that the upper bound on the TTR is approximately $p_1 * p_2$ time steps, which is asymptotical to $O(m^2)$ time. The algorithm performs even better (approximately $O(m)$ time) when the observed channel ordering is the same on both radios.

2.2.3 Sequence Based Rendezvous

DaSilva and Guerreiro propose an alternate rendezvous algorithm in [10] called sequence-based rendezvous. In this algorithm, each radio uses a predefined sequence of channels to visit during the rendezvous process. Sequence-based rendezvous allows for three improvements to the rendezvous process: it has a maximum TTR, and includes a priority order for rendezvous channels.

It is noted that not any sequence can be used to successfully blind rendezvous. The authors argue the problem of finding appropriate sequences is the dual to such coding techniques as frequency-hopping spread spectrum (FHSS). FHSS sequences attempt to minimize the probability of multiple radios occupying the same channel at once, whereas sequence-based rendezvous attempts to maximize this probability. These sequences are both infinite and periodic, with one period of length M. Each of the N available channels is included equally in the sequence, and the algorithm generating the sequences is independent of N.

One method described for producing these sequences is creating a permutation $P(N)$ of the N channels and building the sequence as detailed in Figure 1. The permutation is used $N + 1$ times: N times contiguously, and once interspersed between each of the other permutations. A given example with $N = 5$ is a permutation of (3, 2, 5, 1, 4) [10]. Therefore the sequence would be:

<u>3</u>, 3, 2, 5, 1, 4, <u>2</u>, 3, 2, 5, 1, 4, <u>5</u>, 3, 2, 5, 1, 4, <u>1</u>, 3, 2, 5, 1, 4, <u>4</u>, 3, 2, 5, 1, 4

This sequence would then be repeated indefinitely until rendezvous has occurred.
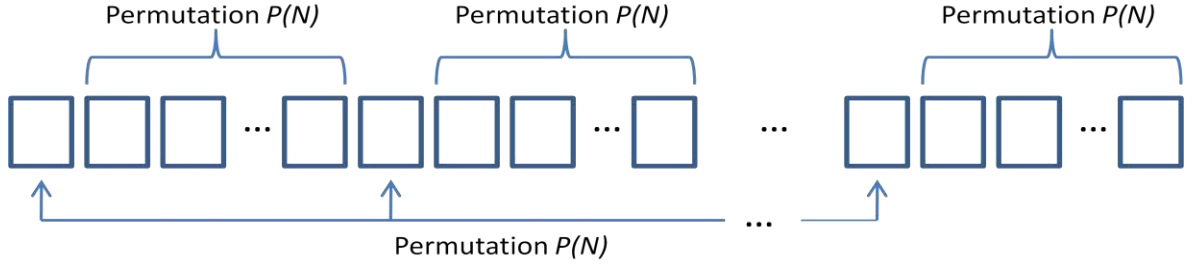
Figure 1: Sequence generated via random permutations [10]

Using these permutation sequences, the TTR for rendezvous is bounded by N^2. Also, some sequences can be generated that favor particular channels despite having equal representation in the sequence. This can be useful by giving priority to certain channels that have higher signal to noise ratio or fewer interruptions. Also, the average TTR for these favored channels is lower than that of random rendezvous. The problem with sequence-based rendezvous is that it requires the radios to have the same available channels and the sequence generation needs to be coordinated.

2.3 Handshaking

In most rendezvous literature, the investigators assume that when two nodes meet in the same channel, they immediately begin communicating. While this simplification is

reasonable for determining the performance of the rendezvous algorithms in arriving in the same channel, it is obviously not very realistic. The two rendezvousing nodes must have some way to know whether they are in the same channel or not, and they must also have some means with which to setup up a connection between themselves. In order for data of any sort (fragments, connection setup, payloads, etc) one of the nodes must be receiving while the other is transmitting. Since both nodes must either be sensing or transmitting whenever they are in a channel, it is possible that they may be receiving or transmitting at the same time and thus would be unable to hear each other, even when in the same channel. This introduces a probability of whether or not this handshake is successful given the two nodes are in the same channel for some amount of time.

2.3.1 Probability of Handshake

In [11] DaSilva denotes this probability of a successful handshake as p_H. The TTR is dependent upon p_H and the expected time for two nodes to be in the same channel at some instant in time. If X is the event that two nodes are in the same channel and the handshake event is uncorrelated with the same channel event, then:

$$P[successful\ rendezvous] = p_H P[X]$$

In the simplest case of random rendezvous, $P[X] = 1/N$, where N is the number of shared channels between rendezvousing radios. DaSilva notes that, in the uncorrelated case, TTR is a 'geometrically distributed random variable representing the number of failures before the first success in a sequence of independent Bernoulli trials with probability of success equal to p_H/N.' Therefore, the probability mass function (pmf) of the TTR is:

$$P[TTR = k] = \left(\frac{p_H}{N}\right)\left(1 - \frac{p_H}{N}\right)^{k-1} for\ k = 1,2,\dots$$

He obtains the expected TTR of N/p_H for random rendezvous. While DaSilva

recognizes that p_H exists, he does not go into detail as to how this probability is

calculated, but states that it is dependent upon three factors: the time spent by each node

transmitting and sensing, and the amount of information a node needs to determine that

another node is in the same channel.

In general search literature, p_H is also acknowledged. Anderson and Weber also

discuss a similar probability they label as α [4] They discuss this as a possible variation

of the search model in which they drop the assumption that if two players search the same

location, then they are sure to find each other. Rather than have a fixed probability for

each location of failing to meet, Anderson and Weber define α as the probability during

some time period that not meeting can occur even if the two players are in the same

location. Therefore, it is possible that this failure to meet probability could occur when

the two players are not in the same location and thus has no effect.

2.3.2 Neighbor Discovery

In [6] Borbash develops an algorithm for neighbor discovery. They look

to maximize the expected number of successful receptions of a message by some

node in the network from one of its neighbors. Each node has its own timeslot,

randomly offset from each other, in which it can either transmit or receive with

some probability. This random offset allows for asynchronous analysis in which

the time slots do not have to be aligned. They define T_m as the time necessary to

send a message, m, and this may be repeated W times. Therefore, each slot is

length $T = WT_m$. A message is successfully heard by a node if it is in the

receiving state, and it receives only one message from any other node. As the

number of possible neighbors increases, so does the chance of collision given the same transmission probability. Therefore, the probability of transmitting changes as number of neighbors changes. Borbash further discovers optimal values for the probability of transmission given certain numbers of neighbors. The algorithm, however, is only interested in receiving a single message to establish a list of neighbors. Therefore, each node only performs one action in each slot, transmit or receive, and does not attempt to respond to any messages. The algorithm is also not concerned with using multiple channels.

Alonso et al. also analyze the node discovery problem for a single channel in [2] however, they elaborate on the talking and listening phases through different protocols. In order to ensure receiving a message in a specific frequency block, only one node can be transmitting and another must be receiving. This also means that if one node receives a message, all other nodes will receive the message, since they must be listening in order for the message to get through. They go on to define the termination of the protocol as when two nodes have found each other: one node successfully sends a message to another node, and that node manages to successfully send a response message to the first immediately. Unlike Borbash's protocol, their proposal requires response messages to be sent; however, the nodes must also be synchronized in that they change states from talking to listening, and vice versa, at the same time.

Alonso then describes five different protocols: random (RP), answering (AP), listen after talking (LP), conditional (CP), and sleep (SP). In the random protocol, nodes choose at random whether to talk or listen, with the probability of

talking as p and the probability of listening as $q = 1 - p$. Using the definition of

protocol termination, the expected time of node discovery with RP is:

$$E[T_{RP}] = \frac{2 - q - p^2}{2p^2q^2}$$

AP focuses specifically on the termination condition by requiring a node

to send a response immediately after receiving a message from another node.

However, this will not work for more than two nodes in the channel. Since all

other nodes would hear the sent message, this protocol would require all of them

to send responses immediately and would thus collide with each other. LP stems

from the fact that a node needs to be able to hear a response after sending its

message. Therefore, after sending a message, a node will listen for responses.

Unlike AP, LP will not immediately respond to messages it receives, so even if a

node receives a message, it is not guaranteed to result in a discovery. The

expected times for node discovery in AP and LP are:

$$E[T_{AP}] = \frac{2 - p^2 - q^2}{2pq^2}$$

$$E[T_{LP}] = \frac{2 - q^2}{2p^2q}$$

CP combines the AP and LP by forcing nodes to both listen after sending a

message and to send a response immediately after receiving a message. For only

two nodes, once a message is heard, the protocol becomes deterministic and node

discovery will complete. The expected time of node discovery with CP is given

as:

$$E[T_{CP}] = \frac{2 - q^2}{2pq}$$

The first four protocols suffer from the problem that for more than two nodes, the responses will collide with each other because more than one node will have heard the original message. Thus, they define SP in which a node will wait for some uniform random time before responding to a message. This is similar in nature to Carrier Sense Multiple Access (CSMA) systems and the back-offs used to avoid collisions. The expected time for SP is (where n is the sleep time):

$$E[T_{SP}] = \frac{1 + (n + 1)pq}{2pq^2}$$

The first analysis with only two nodes showed that CP was better than the others, with SP performing much worse than the other protocols. Further analysis is done for more than two nodes. Because the AP and CP protocols do not lend themselves to more than two nodes in a channel, only RP, LP, and SP are used. The authors determined that SP has the 'best behavior' when the number of nodes is unknown because its performance does not degrade as sharply when moving away from its optimal probability of talking. However, LP performs better on average and is a better choice when the range of the number of nodes is known. Figure 2 shows the optimal performance for SP, LP, and RP, with k = number of nodes. RP is noticeably worse than the others, with LP performing better for the majority of k values.
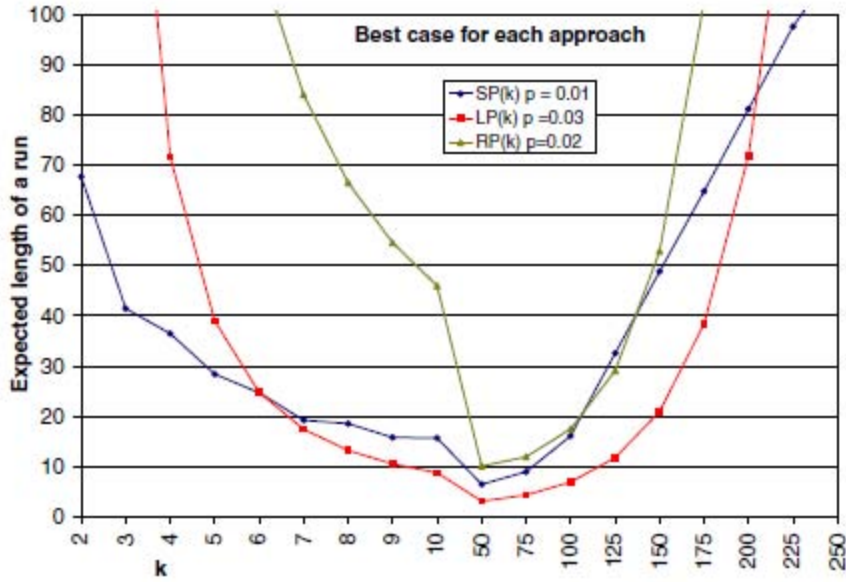
Figure 2: Comparison of the best cases for each protocol [2]

Arachchige investigates the neighbor discovery problem with a specific look at its effects in a Cognitive Radio Network (CRN) [17] They focus on supporting neighbor discovery under asynchronous conditions, similar to Borbash's work. Their symmetric algorithm contains four distinct phases, focusing primarily on the selection of a 'leader node' that then performs the neighbor discovery operations.

The first phase is the Leader Detection Phase (LDP) in which a joining node attempts to find a leader in the CRN. It enters a scanning mode in which it waits for the leader's inquiry messages. If it hears the leader it will respond, otherwise, the new node will attempt to become the leader and enter Leader Election Phase (LEP). During LEP, nodes send messages indicating they wish to become the leader, and eventually one is selected.

19

Once a leader is elected, the Neighbor Discovery Phase (NDP) begins.
The leader will alternatively send out inquiry messages on all its available
channels and listen for responses. Other nodes will respond to the inquiries in a
random time slot based on the available channels and how long the leader is
transmitting for. The responses contain the nodes ID as well as its available
channels. If the leader receives a response, it will finalize the discovery with an
acknowledgement message. The random response is similar to MAC protocols to
avoid collisions between multiple nodes. After the initial discovery phase, the
nodes enter the Normal Operations Phase (NOP). During this time the leader will
periodically restart the NDP in order to update available channel information and
to discover new nodes that have joined.

The inquiry messages and replies are an effective method for two or more
nodes to discover each other and rather than just neighbor discovery, could be
translated to connection setup. However, in the rendezvous domain, especially
with only two nodes, selecting a leader node is impractical. Also, having a leader
node is similar to using a single control channel to perform these operations as it
becomes a single point of failure; however, in this case, a new leader can be
elected.

2.3.3 Multi-Channel MAC Rendezvous

Silvius extends Borbash's work in [21] by applying the algorithm to
multiple channels. The paper is focused on three development areas in
rendezvous: multiple-channel analysis, asynchronous timing, and varying-width
rendezvous slots. Multiple channel cases are similar to single channel cases with

the obvious exception that nodes have the probability of not being in the same channel.

Rather than further derive the mathematical expressions from Borbash's work, Silvius uses experimental procedures to approximate the values. The results were that as the number of channels increases, the expected number of successful messages received decreased. However, the probability of transmitting for optimal performance (to avoid collisions) increases with the number of channels. He also found that increasing the number of repetitions in each timeslot increased the number of expected successful receptions. Finally, in conjunction with the first observation, as the number of channels increased the time to establish a certain number of connections (i.e., discover a certain number of neighbors) increased. While Silvius successfully extends Borbash's work to multiple channels and even multiple timeslot lengths, the algorithm is still focused on neighbor discovery and does not account for the final requirements needed to setup a communications channel.

2.3.4 TCP 3-way Handshake

When discussing handshaking protocols, the famous three-way handshake from transmission control protocol (TCP) is often thought of. When a TCP connection is first made, a client sends a SYN packet to a server to begin connection setup and start the handshake. The server responds with a SYN/ACK which acknowledges the request to start a connection and that the server wishes to connect to the client. The client responds with a final ACK message, and the handshake is complete and connection started [23] .

Typically, when creating a TCP connection, the client has knowledge of the server it is connecting to such as ports and IP address. This means that the client does not

have to spend an extensive amount of time searching for the server. The TCP three-way

aspect of the handshake may be applicable to blind rendezvous, but the assumptions of

prior knowledge and a relatively static server do not apply well.

2.4 Jamming Techniques

While cognitive radios can help improve communications, they can also be used

to disrupt communications. As wireless networks have become more widely used, the

prevalence of malicious activity against these networks has increased. Many of these

activities can be mitigated through proper security practices; however, radio interference

and jamming are not so easily deterred.

2.4.1 Cognitive Jamming

Echo Ridge has researched the fundamentals behind the development of cognitive

jamming [8] They discuss definitions, why cognitive jamming is important,

comparisons to traditional jamming, performance measurement, and enabling

technologies.

Echo Ridge defines cognitive jamming as follows:

"Cognitive Jamming is the impairment of adversarial functionality via an
adversary wireless node physical layer to include direct physical layer attack, and
indirect control plane and user plane attack through the physical layer."

Parts of the definition are kept vague on purpose such as "impairment" and

"functionality" because they include a wide variety of objectives that a cognitive jammer

could be capable of.

When discussing why cognitive jamming is an important research area, they give

four reasons:

- Emerging threat – new capabilities available for electronic attacks

- Application convergence – multiple application wireless devices

- Wireless access in cyberspace – increasing and becoming more available

- Total spectral dominance vision – cyberspace superiority

Echo Ridge has divided up the typical seven layer OSI model into three different sections. The Physical layer remains the same. The control plane consists of the presentation, session, transport, network, and data link layers. The user plane is the application layer. The reason the control plane consists of those five layers is because the distinction between them is irrelevant and/or inaccurate in air interfaces. The "direct attack on the physical plane" refers to what traditional jammers would accomplish by attacking the RF medium. The "indirect attack on the control and user plane" is due to the fact that the attacks must still pass through the physical layer in order to disrupt these planes.

They then present an updated definition for a cognitive jammer as a system of devices capable of performing cognitive jamming:

> "A cognitive jammer is a system of networked nodes that can perceive multilateral situational knowledge related to electronic warfare conditions, then autonomously plan, decide and act on these conditions through an intelligent processing means to achieve end to end goals. The system can learn from these adaptations and use them in future decision making processes."

Key distinctions in this definition are the networked nature of the jammer, the perception of environment, the autonomous nature and adaptability that is associated with a cognitive entity, and the pursuit of end-to-end goals.

2.4.2 Jamming Methods for Wireless Networks

In [29] Xu et al. discuss different jamming techniques and their effectiveness. The two metrics they use are a Packet Send Ratio (PSR) and a Packet Delivery Ratio

(PDR). The PSR measures how effective the jamming was at preventing the sending

node from sending packets out. The PDR measures how effective the jamming was at

preventing useful packets from reaching the receiving node. Each of their jamming

methods affects these in different ways.   They discuss four different types of jamming

attack models: the constant jammer, the deceptive jammer, the random jammer, and the

reactive jammer.

The constant jammer continuously sends out noise in the channel that two nodes

are communicating in. Because this jammer is constantly sending out signals, it is costly

in terms of power consumption, and can also be easily identified and possibly taken out.

They mention that if there is a small threshold that the MAC protocol uses to determine if

the channel is idle, then the jammer may only need to put out that much noise in order to

tie up the channel.

The deceptive jammer is similar to the constant jammer except that instead of

continuously broadcasting noise, the deceptive jammer sends out legitimate packets. The

idea is that it will deceive the other nodes into believing that legitimate traffic is being

sent and thus they will be unable to send their own. This assumes that the deceptive

jammer knows the packet format being sent.

The random jammer can behave as either a constant or deceptive jammer, but

only during certain periods. The random jammer will randomly sleep for certain time

periods and then jam during others in order to make it less discoverable.

The reactive jammer is different than the first three, which would be considered

active jammers. The reactive jammer, instead of being active regardless of the targets'

actions, will wait until it hears transmissions being sent by the target and then activate its

24

jamming. The first three jamming methods affect both the PSR and PDR. Reactive jamming, however, does not affect the PSR because the target needs to have sent a packet in order for the reactive jammer to detect it.

This paper analyzed the various methods and their effects on PSR and PDR at varying distances from the target. According to them, the most effective jamming was the deceptive jamming because the seemingly legitimate packets cause the targets to constantly stay in reception mode instead of sending anything. The deceptive jamming blocked both the sending and delivery of packets. The constant jammer was the next effective jammer; it was not as effective against preventing packets from being sent, but the noise generated managed to corrupt packets before delivery, thus reducing the PDR. The random jammer was minimally effective against PSR, and fairly effective against the PDR at the closer distances. The reactive jammer was mostly ineffective against PSR for reasons discussed earlier, but at close distances managed to completely disrupt packet delivery.

One limitation is that these methods were used against a single channel using well-known/documented wireless protocols that were unencrypted. In order to incorporate more channels, a channel hopping algorithm would need to be developed.

2.4.3 Jamming 802.11 Networks with Cognitive Radios

Sampath et al. [20] discuss how cognitive radios can be used to jam 802.11 networks. They discuss three features of cognitive radios that allow them to jam spectrum efficiently: real-time spectrum sensing, fast channel switching, and software-reconfigurability. They analyze jamming both single and multiple channels using a cognitive radio. The simplest implementation of jamming a single channel is to constantly transmit noise into the channel; however, because this is easy to detect and

expensive, they opt for periodic jamming. They argue that jamming effectiveness is a function of the jamming intervals, the jamming packet size, and the legitimate packet size being sent to the victim. Their results showed that the jammer effectiveness decreased as the jamming intervals increased, and that the larger victim packets were more vulnerable to jamming. Since a victim can avoid single channel jamming by switching channels, the authors next examine multiple channel jamming with a single cognitive radio.

They ran simulations using Qualnet 3.8, a network/discrete event simulator. Using cognitive radio channel switching capability combined with the advanced channel sensing, cognitive radio jammers can quickly switch to channels that are more likely to harbor legitimate traffic and jam them. They measured the jammer performance by the percentage of user traffic corrupted by the jammer on each channel. Specifically, they looked at the impact the number of channels, jamming packet size, and channel switching delay had on the jammer effectiveness. As the number of channels increased, the jamming effectiveness decreased. Smaller jamming packets were the most effective because only a small amount of information is necessary to corrupt legitimate packets and the smaller size decreased the amount of time spent jamming a particular channel. The lower switching delay also increased the effectiveness of the jammer. The significance of this research is that a single cognitive radio can be used as an effective jammer against multiple channels.

2.4.4 Security-enhanced Virtual Channel Rendezvous Algorithm (SVCR)

The SVCR is an algorithm designed by Ma and Shen in [18] to be more robust against jamming nodes, especially 'smart' jammers. The algorithm focuses on re-rendezvousing after being jammed as opposed to avoiding jamming during the initial rendezvous process. They highlight three major vulnerabilities in current rendezvous

algorithms (common control channel, predictability, and observability) and aim to limit them with their algorithm.

They remove the first vulnerability by simply not using a common control channel, essentially creating a blind re-rendezvous problem. The second is solved through pseudorandom channel mapping. In this implementation, each radio's available channels are mapped to pseudorandom values based on the number of available channels and a random seed which is pre-programmed into each node. The pseudorandom sequence generated is finite and static, so eventually a smart jammer could discover it, however it would take an exceptionally long time. To defend against the final vulnerability, they implement virtual channels using direct spectrum spread sequences (DSSS). DSSS involves adding a noise factor into the signal to make it appear like white noise. The noise is then factored out of the signal at the receiving end. DSSS requires a certain amount of synchronization between nodes to work. While these measures protect against jammers at the signal level, the messages themselves are still in the clear assuming a jammer can decipher the signal. Therefore, they recommend encrypting the messages using common cryptographic techniques such as AES, SHA256, and public key authentication.

While their analysis takes place after the initial rendezvous, it can still be applied before rendezvous occurs. Rendezvous algorithms often utilize some form of randomness when choosing channels, which helps eliminate predictability. Methods similar to DSSS can also be implemented into the rendezvous process while still maintaining symmetric characteristics. Likewise, encryption of rendezvous messages can be implemented which can protect against eavesdropping and deceptive jammers.

III. Modeling and Jamming the Rendezvous Handshake

Wireless rendezvous and jamming techniques for cognitive radios are a relatively new area of research, particularly when the rendezvous process must be concerned with a radio that is intentionally jamming the rendezvous process. This chapter discusses the key components of jamming two radios attempting to rendezvous. First, it describes a mechanism for performing the handshake when two cognitive radios are in the same frequency band at the same time. Next, it describes four rendezvous jamming algorithms: a noise jammer, a deceptive jammer, sensing jammer, and a primary user emulation jammer. It then identifies key metrics in assessing the performance of these algorithms. Finally, an experimental design is developed to test the performance of these algorithms.

3.1 Handshaking

To allow two radios to detect each others presence once they have arrived in the same channel, a handshake is developed. The handshake is modeled after the work done by Borbash and Alonso. This handshake is not guaranteed to be successful, and as such, introduces a probability that two radios may not rendezvous even when in the same channel.

3.1.1 Handshake Model and Algorithm

We characterize a handshake by the following process: an initial beacon from one radio is heard by another; the receiving radio responds to that beacon, which is then heard by the initial radio; and a final response by the initiating radio is transmitted and heard by the receiving radio. The handshake process works such that when a radio enters a channel, it performs up to three different actions (depending on how successful the previous actions were): beacon *transmission*, beacon *receiving*, and beacon *response*.

Thus when a radio enters a channel for the first time, it transmits one or more beacons and then listens for a response. Should a radio hear a beacon, it responds with a special response beacon. If a radio does not detect any beacons during it's receive time, it moves to a different channel. Finally, if the handshake algorithm does not receive any useful beacons or responses for some amount of time, it will timeout and switch channels. This timeout could occur from a variety of reasons such as the other radio dropping its connection or the presence of a jammer.

Alonso's work in [2] is most similar to this handshake in that he required messages and responses in order for the neighbor discovery protocol to terminate. Also, like his initial protocols, we are only concerned with two radios attempting to rendezvous.

Figure 3 presents an overview of the handshake algorithm. The concept of an initial beacon is similar to the work by Borbash in [6] with the added requirement that a bi-directional connection must also be made, which is why the full handshake algorithm includes response packets.
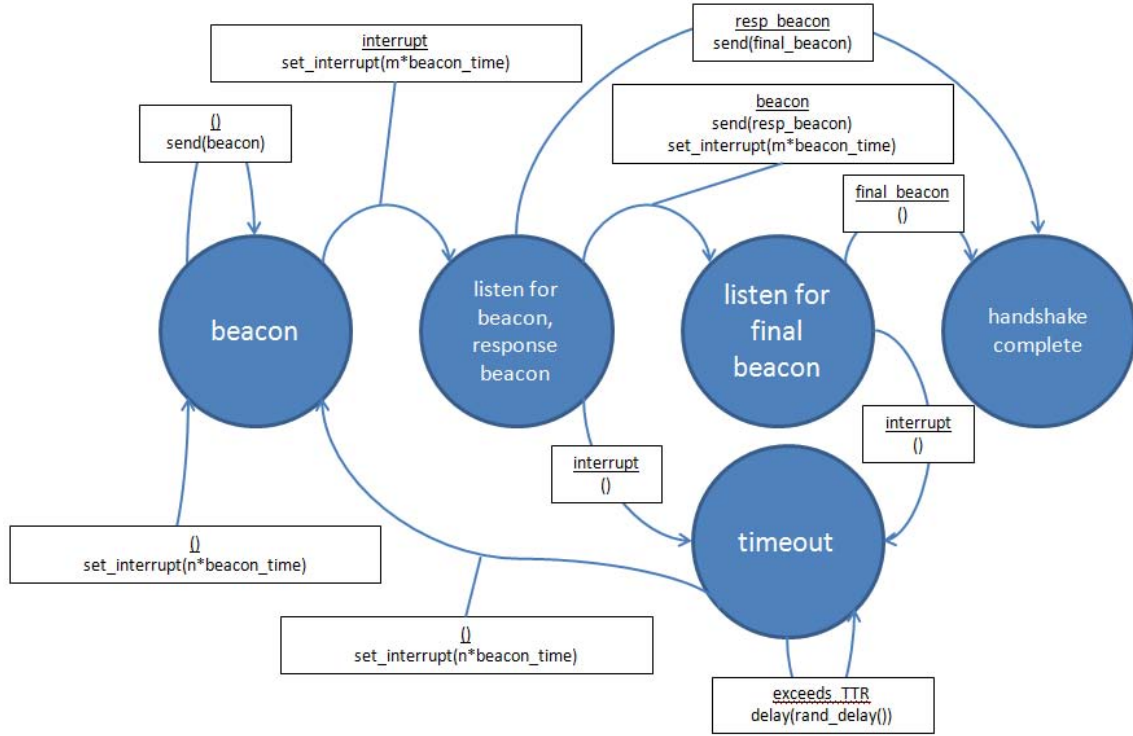
Figure 3: State Transition Diagram for Handshake Algorithm

In Figure 3, each radio can transmit and receive for a certain number of time slots.

Let n be the number of transmit slots used, and m be the number of receive slots, where

the total slots used, $n + m = T$. We assume that a radio transmits before receiving

when it enters a channel and that each radio uses the same strategy (i.e. they use the same

number of transmit and receive slots). Therefore, upon entering a channel, a radio will

transmit *n* beacons. At this point in the state transition diagram (STD), the interrupt will

engage, causing the radio to change states into a receiving mode for *m* timeslots. We also

assume that a radio must receive a beacon in its entirety to fully decode and understand it,

so these time slots are measured in the time it takes to transmit an entire beacon in the

channel, denoted as τ (value given later). It should be noted that m should not be less

than or equal to 1, otherwise the transmission slot will need to line up exactly with the

one receive slot, or the receive slot will not be long enough for full reception. Furthermore, under the condition that $m = 1$, it is extremely unlikely the beacon and single receive slot will align. Should a radio receive a complete beacon during its receiving phase, it will send out a response beacon. If it receives a response beacon to its initial beacon, it will send out a final beacon. If it hears nothing, it will interrupt and change states to a timeout, after which it will change channels and enter the transmitting phase again. After sending a response beacon, the radio will enter another receiving phase, awaiting a final response beacon. If a final response beacon is received, the handshake is completed; otherwise, a timeout will occur and the radio will interrupt, change channels, and begin transmitting again.

We define the absolute offset between the times when the two radios entered the channel as Δ. This offset characterizes the asynchronous nature of the rendezvous, meaning that the radio timeslots do not have to line up perfectly to rendezvous. We assume that Δ is a uniformly distributed random variable, indicating that the possible times any radio can enter a channel are equally likely. We also assume Δ is bounded between 0 and $T\tau$. $T\tau$ is the largest offset possible between two radios while they are in the same channel. While the channel entrance times for each radio relative to each other can range from $-T\tau$ to $T\tau$, because the offset is the absolute difference between the radios' times, we do not use negative values. Figure 4 illustrates the handshake scenario with one radio's transmit being heard by the other. Upon receiving the beacon, the receiving radio immediately sends the response, followed by the transmitting radio sending its final response.
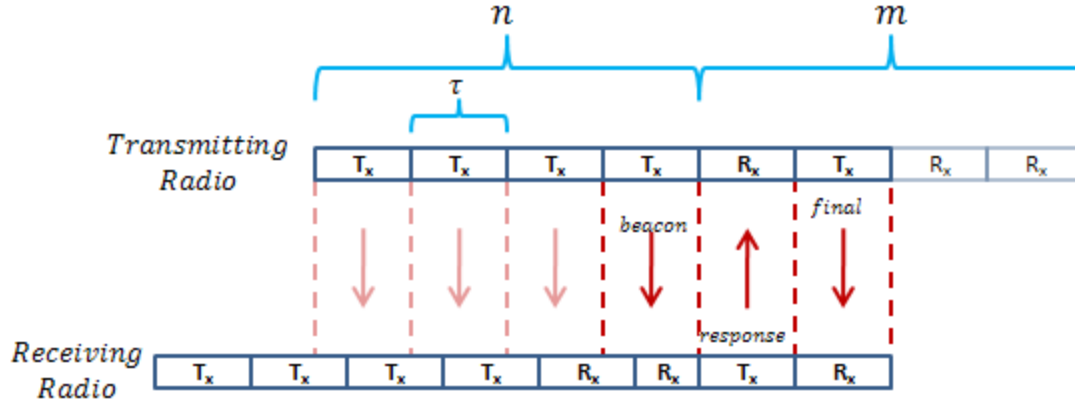
Figure 4: Rendezvous handshake: T_x=transmit slot, R_x=receive slot, note that neither radio has exactly m receive slots in a row because of when the beacon and response were received

3.1.2 Failure Case

An important part of the STD in Figure 3 is the case where the elapsed time has surpassed what we would have expected the time to rendezvous (TTR) to be. The p_H from DaSilva's work discussed in Chapter 2 assumes that the handshake method used is randomized and each subsequent handshake is not correlated with the previous. If the radios use the same handshake parameters every time in the same channel, there are values of Δ when the transmit and receive slots will line up in a way that the two nodes will never successfully handshake unless the handshake is modified in some way. Figure 5 shows how, without propagation delay, two nodes may repeatedly end up in the same channel and yet never accomplish a handshake. If $R2$ enters the same channel as $R1$ in offset A (immediately after $R1$ could have heard $R2$'s beacon), at some point in the future, $R2$ could also enter the same channel as $R1$ with offset B (immediately after $R2$ could have heard $R1$'s beacon). It can be seen that neither of these offsets result in a successful handshake, and unless the offset is shifted, the two nodes will never

32

handshake, resulting in an infinite TTR. Figure 6 shows another case in which no handshake can occur.

This failure is a result of the assumption that a radio must receive a full beacon in order to understand it. Because of this assumption there are values of the offset Δ in which a handshake will fail $\Delta = (0, \tau)$ and $\Delta = (T - \tau, T)$. This gives a 2τ failure window for each radio, totaling 4τ, and will be explored further in Section 4.1.4.

Thus, if a radio's elapsed TTR has surpassed its expected TTR, it will delay itself from entering the next channel to increase the probability the handshake will be aligned properly. However, time slot alignment during the handshake is only one problem that would result in longer TTR. Other causes for surpassing the expected TTR could be the presence of a jammer or using a rendezvous algorithm without an upper bound on TTR.

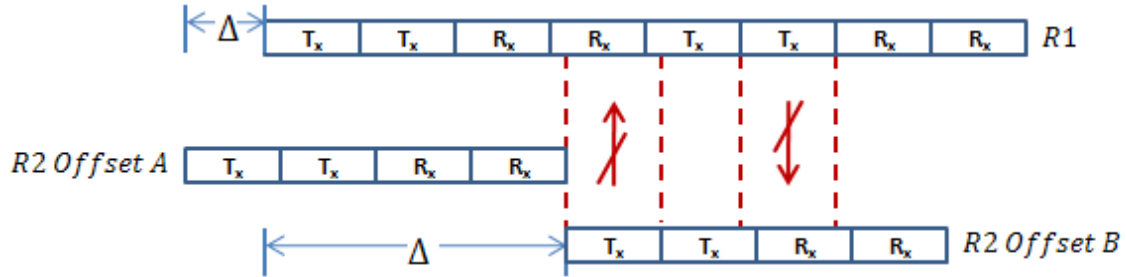
Figure 5: Unsuccessful handshake for both possible offsets, $\Delta \approx (0, \tau)$ and $\Delta \approx (T-\tau, T)$
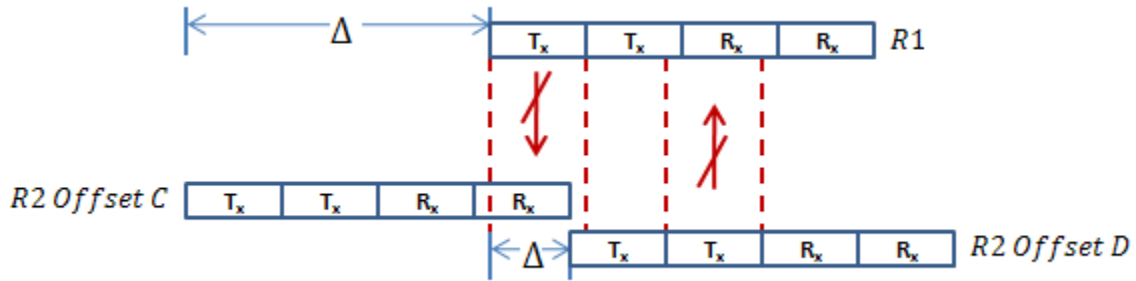

Figure 6: Unsuccessful handshake for both possible offsets, $\Delta \approx (0, \tau)$ and $\Delta \approx (T-\tau, T)$

3.2 Jamming Algorithms

In order to effectively jam the rendezvous process, the jammer must accomplish two objectives: enter the same channel as one or more of the rendezvousing radios and then disrupt their communications. To achieve the first objective, the jammer will essentially need to rendezvous with the rendezvousing radios. Therefore, using the current *best* rendezvous algorithm would net the jammer the highest likelihood of entering into the same channel as one of rendezvousing radios. We will use a random strategy for the jammer's rendezvous strategy as a baseline.

Once the jammer is in the same channel as a rendezvousing radio it can either broadcast noise or can attempt to deceive the rendezvousing radio into believing the jammer is a legitimate radio. For all jammers, we assume correct functionality; we do not detail specific message formats for deception or primary user emulation, nor do we determine how much noise needs to be transmitted. The effectiveness of these algorithms will be analyzed in Chapter 4.

3.2.1 Noise Jammer

The first jamming strategy is to broadcast noise into the channel the jammer enters to disrupt any communications in that channel. This is the same as the constant jammer described in Section 2.4.2 Jamming Methods for Wireless Networks. We refer to it as a *noise jammer* because it better differentiates it from the deceptive jammer. This strategy is very effective against established communication channels, albeit easy to detect and power-hungry; however, it is not as effective before the radios have established their connection. When a rendezvousing radio enters a channel, it is looking for specific packets to know that another radio is present. If the radio only hears noise, it

34

will assume that the radio it is looking for is not present and will switch channels. Broadcasting noise into a channel with a single radio has virtually no effect on disrupting the rendezvous process. The only time noise will be effective is when both radios are in the same channel handshaking. In this case, noise will keep them from receiving each others' beacons and force them to rendezvous again. Figure 7 shows a state transition diagram for the noise jammer. The interrupt is defined as the condition under which the jammer will change states (i.e., once it has transmitted noise for a certain length of time). In our implementation, each jammer spends the same amount of time in each channel as the rendezvousing radios ($n + m$ time slots). We assume the time it takes for the jammer to change channels is negligible, but it will still halt transmitting while changing.
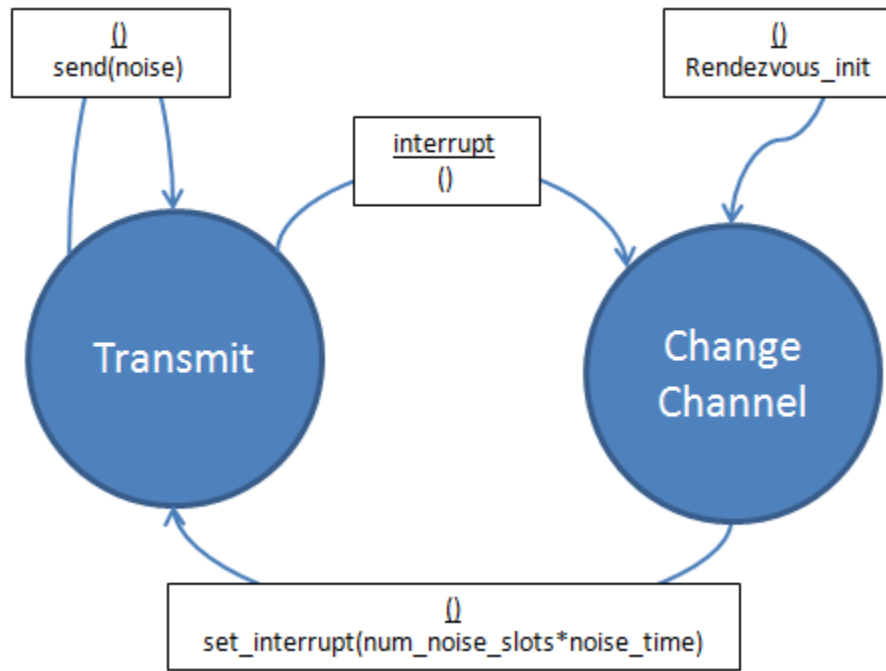

Figure 7: State Transition Diagram for Noise Jammer

3.2.2 Deceptive Jammer

The second jamming strategy involves sending real packets in the channel rather than noise, like the deceptive jammer discussed in Section 2.5.1. This jammer will mimic the rendezvousing radios in that it will send beacons as soon as it enters the channel but will not bother listening for a response. The goal of this jammer is to keep the rendezvousing radio occupied and unable to rendezvous with the other radio. In [29] Xu showed that the deceptive jammer was able to keep communications tied up indefinitely. In our implementation, we assume that the rendezvousing radios will time out after not receiving any useful information for some period of time and then move to a different channel. Therefore the best this jammer can do is delay the rendezvous. Should this jammer enter a channel with both rendezvousing radios, it is possible that a collision would occur depending on the signal strength of the jammer and other radios. It is also possible that rendezvous would still occur. We therefore assume in our model that this jammer will not affect two rendezvousing radios in the same channel (thus making our performance estimate conservative). Figure 8 shows the STD for the Deceptive jammer.
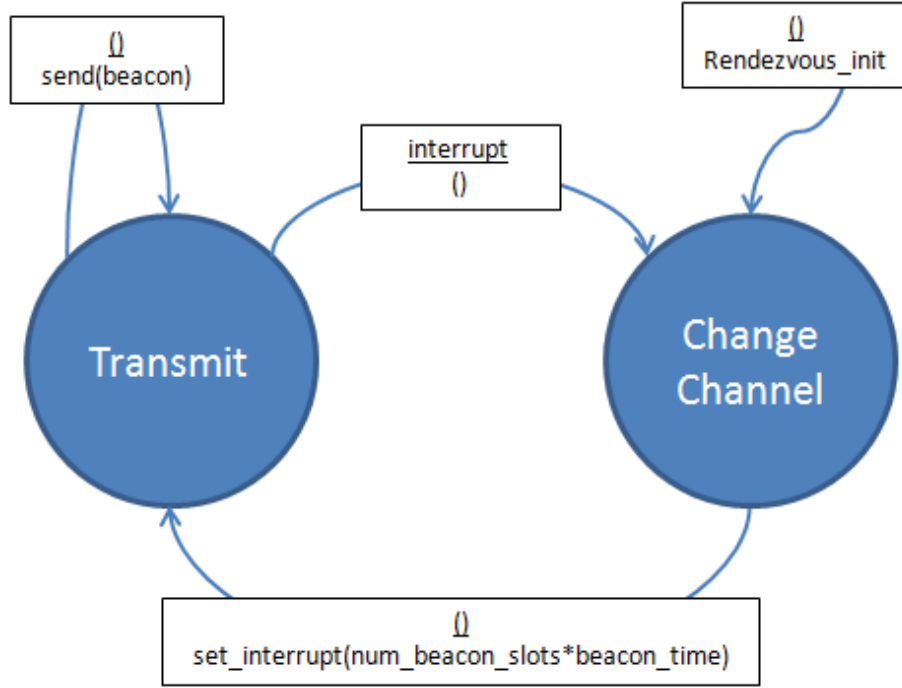
Figure 8: State Transition Diagram for Deceptive Jammer

3.2.3 Sensing Jammer

The third jamming strategy is a hybrid of the first two, similar to the reactive jammer described in Section 2.5.1. We refer to it as a sensing jammer to differentiate it from Xu's reactive jammer, because the reactive jammer begins jamming as soon as it hears any activity in the channel, whereas ours waits until it specifically receives beacons or responses before jamming. Instead of sending beacon packets as soon as it enters the channel, it will listen for other packets being sent. If it hears a beacon, it can send a response and occupy the rendezvousing radio (in a manner similar to the deceptive jammer). If it hears a response, it would know that the two rendezvousing radios are attempting to handshake and can then broadcast noise into the channel and make them rendezvous again. Figure 9 shows this functionality in a state transition diagram.

37
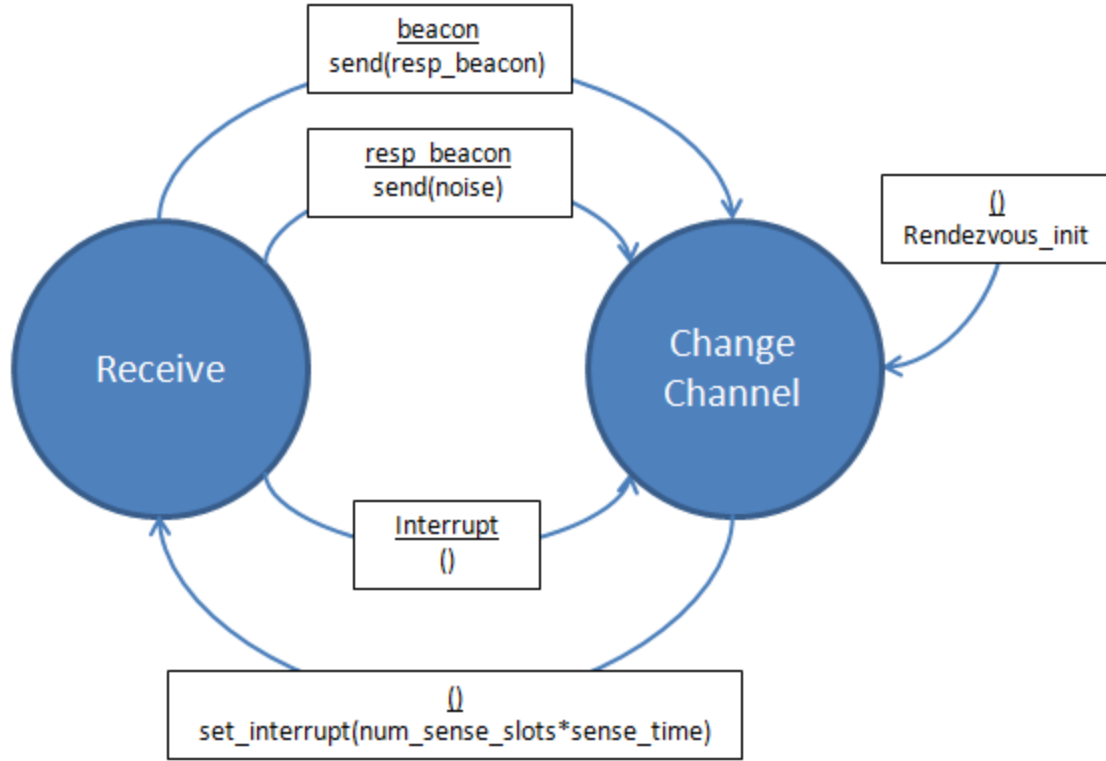
Figure 9: State Transition Diagram for Sensing Jammer

3.2.4 Primary User Emulation Jammer

The final jamming strategy would be to transmit a primary user signal into the channel (shown in Figure 10). If the jammer transmits this signal when a rendezvousing radio is in the same channel, the rendezvousing radio would recognize the channel as being used by a PU, and thus would not return there during the remainder of the rendezvous process. This would reduce the number of common channels available to the rendezvousing radios, which would reduce the probability of rendezvous. Similar to the deceptive jamming, primary user emulation (PUE) assumes the jammer is aware of the packet format and signal structure to emulate. If a PUE jammer catches both radios in the same channel at once, it is assumed that it will affect both simultaneously, causing both radios to remove that channel from their list.
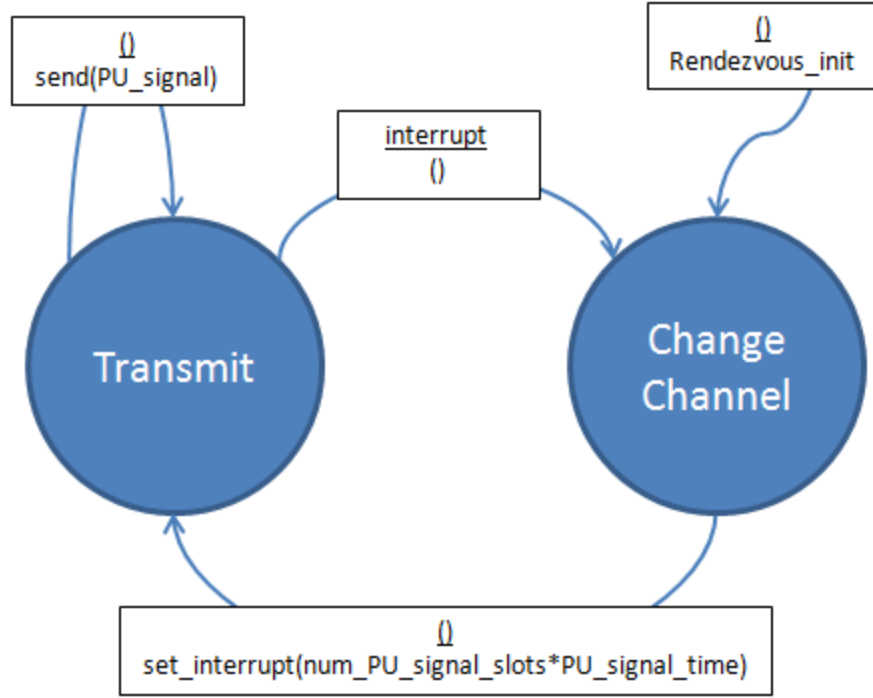
Figure 10: State Transition Diagram for PUE Jammer

3.3 Metrics

To measure the performance of the handshake and jammers, the most critical question to understand is how much they increase the expected rendezvous process completion time. Internal to the handshake process, we want to know the probability of a successful handshake and how long it is expected to take to complete a handshake. We are also interested in how long it takes each jammer to successfully jam, and the probability of successfully jamming before a successful rendezvous occurs.

3.3.1 Rendezvous Metrics

To analyze the effect of the handshake and compare the jamming strategies, the most obvious metric is the increase in expected time to rendezvous (TTR) after introducing the handshake/jammer over the time to meet (TTM). TTR is measured as

time that elapses from when both rendezvousing radios are actively trying to rendezvous with each other and when the rendezvous has actually occurred.

$$TTR = t_r - \max(t_1, t_2) \qquad (1)$$

Where t_r= time of rendezvous, t_1= time R1 starts, t_2= time R2 starts.

The expected time to meet (E[TTM]) measures the time it takes for two radios to enter the same channel, meaning that once two radios are in the same channel at the same time they have met. The TTM is calculated from when both radios have begun trying to rendezvous to when they meet in the same channel (but do not necessarily rendezvous), as well as between meetings (they may meet several times before successfully rendezvousing). The TTM is different from TTR in that the handshake is not required; however, the TTM is still affected by the number of transmit and receive slots in our implementation because they affect how long before a radio changes channels. The TTM, when compared to the TTR, provides insight into the effects of the handshaking algorithm. We can calculate E[TTM] as

$$t_{m(0)} = \max(t_1, t_2)$$

$$E[TTM] = \sum_{i=j}^{M-1} t_{m(i+1)} - t_{m(i)} M \qquad (2)$$

where $t_{m(i)}$ = time of ith meeting and $M > 0$ is the number of meetings before rendezvous.

3.3.1 Jamming Metrics

A specific jamming metric is the probability of jamming (p_J) of the jamming algorithm. This represents the probability that, given the jamming radio starts at the same time as the rendezvousing radios; it either delays or disrupts rendezvous. The probability of jamming is calculated from the number of successful jams before rendezvous out of the total number of steps (channel changes) the jammer performed before rendezvous.

Since different algorithms are designed to do one, the other, or both, this will result in varying probabilities of successful jamming. For example, the noise jammer's effectiveness is affected by the fact that when it is in the same channel as only one radio, it is ineffective. The deceptive jammer, however, is affected more by whether or not the jamming packets are heard by the radio, similar to the handshake.

The expected Time To Jam (E[TTJ]) is also measured. The TTJ is measured beginning when the two radios begin trying to find each other and ends when the jammer has effectively jammed the rendezvousing radios. The TTJ is also measured from the end of one jam until the beginning of the next successful jam. We take the average of these times to determine the E[TTJ].

$$E[TTJ] = \frac{\sum_{i=0}^{N-1}(t_{j(i+1)} - t_{j(i)})}{N} \qquad (3)$$

Where $t_{j(i)}$ = time of ith successful jam, $t_{j(0)} = 0$ and there are $N > 0$ successful jams.

Another important metric for the jammer is the expected duty cycle (E[DC]). The E[DC] is the percentage of time the jammer is expected to be performing its function. The E[DC] for the noise, deceptive, and PUE jammers is 100% because they are constantly transmitting either noise or fake packets into whatever channel they are currently in. The sense jammer, however, will have an E[DC] less than 100% because it will be sensing the spectrum until finding a reason to transmit. The $E[DC]_{sense}$ is a function of the number of successful jams before rendezvous occurs and the expected amount of time spent jamming.

$$E[DC]_{sense} = \frac{\# \, successful \, jams * E[t_{transmitting \, jam \, signal}]}{TTR} \qquad (4)$$

Where $t_{transmitting \, jam \, signal}$ is the amount of time the jammer spends transmitting the jamming signal.

3.3.2 Handshake Metrics

The probability of a successful handshake is a function of the number of transmit and receive slots, as well as propagation delay. This is the probability that, given a uniform distribution of offsets and that two radios are in the same channel at the same time, the handshake is successful. This is further analyzed in Section 4.1.4.

Perhaps the most important metric of the handshake process is how long it takes to complete once it begins. In the simplest sense, the time to handshake starts when the radios enter the same channel and ends upon a successful handshake. However, it is also a function of the transmit and receive slots used, as well as propagation delay and when each radio entered the channel. A detailed explanation is given in Chapter 4.

$$TTH = t_h - t_m \tag{5}$$

where t_h = time of successful handshake, t_m = time radio receiving the initial beacon began handshake.

The handshake completion time directly affects the time to rendezvous. The expected time to successfully rendezvous is a function of the expected time to meet, E[TTM], and the expected time to successfully handshake, E[TTH], as well as the probability of successfully handshaking when meeting. On average, it will take $1/p_H$ times for the handshake to be successful, thus the two radios will have to meet that many times before rendezvous occurs. We then add the time for the successful handshake, and assuming the handshake and meetings are independent we obtain Equation (5).

$$E[TTR] = \frac{1}{p_H} * E[TTM] + E[TTH] \qquad (6)$$

The longer two radios remain in the same channel attempting to handshake

increases the chances that a jamming radio will discover them and force the rendezvous

process to start over. Once rendezvous has occurred, the radios can agree on fallback

channels in case communication is interrupted, but until then the radios want to minimize

the time spent handshaking, and maximize the probability of the handshake successfully

completing.

These metrics will be evaluated under different factor conditions. For example,

analyzing the effect of delay or different time slots would include determining the

increase in TTR or a change in the probability of rendezvous.

3.4 Performance Evaluation

There are two objectives in the evaluation of the handshaking and jamming

performance. First, the handshaking performance is evaluated by determining p_H and

$E[TTH]$. Second, the relative performance of the jamming algorithms is investigated. To

test the validity and effectiveness of the jamming strategies, they are tested under

published rendezvous algorithms. These include the random and modular clock

algorithms. To analyze the effect of the new handshake method, various values of *n* and

m (transmitting and receiving times) are used. The interaction between the rendezvous

schemes, handshaking parameters and jamming algorithms is then investigated.

3.4.1 System Parameters

There are many parameters that affect the metrics of this experiment. An

important parameter is the number of channels available to the rendezvousing and

jamming nodes. For random rendezvous the probability of a radio being in any one of *N*

channels is 1/*N*. Because the probability is inversely proportional to the number of channels, the TTM increases as the number of available channels increases. Therefore, as the number of channels increases, the probability of either rendezvousing or jamming is likely to decrease when using other rendezvous algorithms.

Two parameters that have an effect on delay, specifically transmission delay, are the size of the beacon and response packets and the transmission rate of the radio. The beacon and response packets are set to 16 bytes, which is similar to the 20 byte request to send (RTS) packets in 802.11 [14] The bit rates for the two protocols above are 2Mbps and 14Mbps respectively. For simplicity, we will use a transmission rate of 10Mbps. This gives a transmission time for one beacon/response of 0.0128ms. Thus, we will set τ to 0.0128ms, for all analysis.

Another parameter is the propagation delay, *d*, which is a function of the distance between the two nodes. Various wireless protocols are in use today, and two of the most prominent in use for local and metropolitan areas are 802.11 WLAN and 802.16 WMAN. The authors of [16] develop co-existence algorithms for both these protocols using cognitive radios. During their experiments they give maximum coverage ranges and raw bit rates for both 802.11 and 802.16 given certain receiver and transmitter parameters. The ranges given for 802.11 and 802.16 are approximately 550m and 3km (respectively). These ranges are used to generate propagation delays of 1.83us and 10us for simulation and analysis on our handshake and jamming algorithms.

In most cases, the propagation delay will be negligible. In Chapter 4 we will show that the propagation delay becomes an important factor once *d* is close to the time slot length, τ. Therefore, the propagation delays we have chosen are near the same value

as our τ. As an example, the worst case propagation delay likely to be experienced would be from communicating via a geosynchronous satellite. This would results in a distance of 35, 786km from the radio. Using speed of light transmissions, it would take 0.1193s for a message to travel from the radio to the satellite, and vice versa. Assuming a constant message size of 16 bytes, if we used a transmission rate of approximately 1Kbps we would achieve a transmission delay similar to the propagation delay, at which point the propagation delay would have significant effects on the handshake.

3.4.2 Factors

Most of these parameters are either fixed at a particular value, or assumed to be negligible for the particular aspect of the system being studied. Seven parameters are varied during the experiment, making them factors. These are the rendezvous algorithms, the jamming algorithms, the number of transmit and receive slots, the propagation delay, the number of available channels, and the backoff time ranges when the radios timeout. Table 1 lists all the factors and their possible levels.

Table 1: Factors and Levels

Factor	Levels
Rendezvous algorithm	Random Modular Clock Algorithm
Jamming algorithm	Noise jammer Deceptive jammer Sense jammer PUE jammer
# Transmit slots	1, 2, 3, 4
# Receive slots	2, 3, 4, 5
Propagation Delay	0, 1.83us, 10us, 12.8us, 19.2us, 25.6us
Total Channels	5, 10, 25, 50
Timeout Backoff delay	2τ, 3τ, 4τ, 5τ, 6τ, 7τ, 8τ, 9τ, 10τ

3.4.3 Evaluation Technique

Of the three evaluation techniques available, the analytical and simulation techniques are chosen for this experiment. Analysis can be used to determine bounds on performance. Furthermore, since both the jamming and handshake communication methods are a relatively new areas of research, an analytical approach provides a good baseline from which to evaluate either a simulated or measured response. An analytical approach may produce simplified equations which can be applied using the simulation factor levels. Simulation is chosen to examine cases where analytical solutions are less tractable and determine variance. Using previous simulation implementations of algorithms (random, modular clock), the radios can be modeled to utilize the various rendezvous algorithms and handshake method. Jamming functionality is then implemented onto a virtual radio node and the effects analyzed. The analytical results will be used to validate the simulation data obtained.

3.4.4 Simulation Design

Different factors will be useful in analyzing the performance of the handshaking and jamming algorithms. To analyze the handshake, we will use both rendezvous algorithms. All transmit and receive slot variations will be used, as well as six different propagation delays. Various backoff delay ranges will be used for when the radios timeout.

For analyzing the performance of the different jamming strategies, both rendezvous algorithms will be used, as well as all four jamming algorithms. Four combinations of transmit and listen slots are also used to analyze effects the handshake

may have on jamming. Four different numbers of channels are also used. The

rendezvous algorithm used by the jammer is set at random. Figure 11 shows the various

factors being used for each analysis.

In order to consolidate the large amount of data from running 100,000 simulations

algorithms, the simulations are able to be performed in Matlab 7.4.0 running on Windows

XP SP2. Each configuration was run 100,000 times. Data was output to text files as the

program ran, which were then transferred into Microsoft Excel and later to Minitab 15.

In order to consolidate the large amount of data from running 100,000 simulations

of each configuration and to produce normally distributed data, we invoked the Central

Limit Theorem, which states that the sample means from a set of independent and

identically distributed random variables tend towards the normal distribution [22]   This

allows us to take small samples from the 100,000 runs, average them, and use those as

the data points. These means will tend toward the overall mean and will produce a

normal distribution. Therefore, during simulation, we average every 100 runs and use it

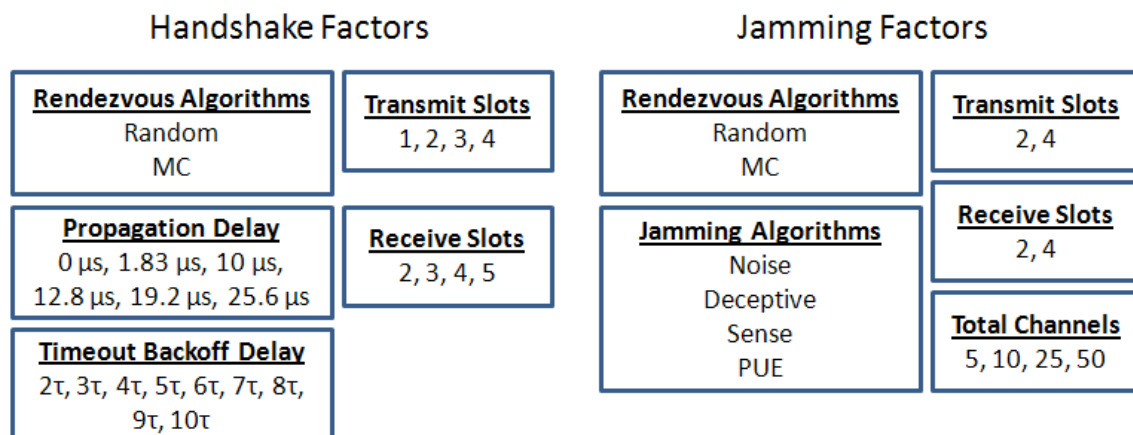as a data point, leaving us with 1000 total data points per configuration.

Handshake Factors   Jamming Factors

Handshake Factors			Jamming Factors		
Rendezvous Algorithms Random MC	**Transmit Slots** 1, 2, 3, 4		**Rendezvous Algorithms** Random MC	**Transmit Slots** 2, 4	
Propagation Delay 0 μs, 1.83 μs, 10 μs, 12.8 μs, 19.2 μs, 25.6 μs	**Receive Slots** 2, 3, 4, 5		**Jamming Algorithms** Noise Deceptive Sense PUE	**Receive Slots** 2, 4	
Timeout Backoff Delay 2τ, 3τ, 4τ, 5τ, 6τ, 7τ, 8τ, 9τ, 10τ				**Total Channels** 5, 10, 25, 50	

Figure 11: Different factors used in measuring the effectiveness of the handshake and jammers

IV. Analysis

This chapter will detail the results obtained through mathematical analysis and simulation. We begin by looking at the mathematical analysis of the handshake algorithm, followed by the simulation results for the same, and then compare the two. Next, we discuss the simulation results for the various jammers used and their effects on the rendezvous and handshake. Finally, we compare the effectiveness of each of the jammers.

While the simulations contain many different parameters, unless otherwise specified, the default values for the factors are:

Table 2: Default values for various factors

# Channels (N)	25
Propagation Delay (d)	0s
Transmit Slots (n)	4
Receive Slots (m)	4
Time Slot Length (τ)	0.0128ms

4.1 Mathematical Handshake Analysis

4.1.1 Simple Case Analysis

We begin by looking at a simple case with our default values, except that the transmit/receive slots have been changed to $m = 2$ and $n = 2$; two transmit slots followed by two receive slots with no propagation delay. The two radios used are denoted as $R1$ and $R2$. Figure 12 illustrates the range of possible time slots in which any two radios would be in the same channel for at least some amount of time, where time is referenced from the perspective of $R1$. For this example, the range of time slots that two radios will be in the same channel is from -4τ to 4τ (for a total of 8τ) of possible

offsets. Inside of this 8τ range, there is also a range in which either radio can successfully receive an initial beacon from the other.

Figure 13 shows the range of possible time slots that $R2$ can receive transmissions from $R1$. The earliest $R2$ can receive a beacon is if $R1$'s first transmitting slot lines up with $R2$'s last receiving slot. The latest $R2$ can receive a beacon is if $R1$'s last transmitting slot lines up with $R2$'s first receiving slot. This gives a total range of 2τ. Figure 14 is similar to Figure 13 except that $R1$ is receiving the beacons from $R2$. This also gives a total range of possible alignments for $R2$ to successfully receive $R1$'s beacon of 2τ. Thus, we have a combined time period of 4τ (with no overlap) in which either radio will receive the other's beacon out of a total time range they could be in the same channel of 8τ. This gives a 50% probability of successful beacon reception given all alignments are equally likely and they are in the same channel for some period of time.
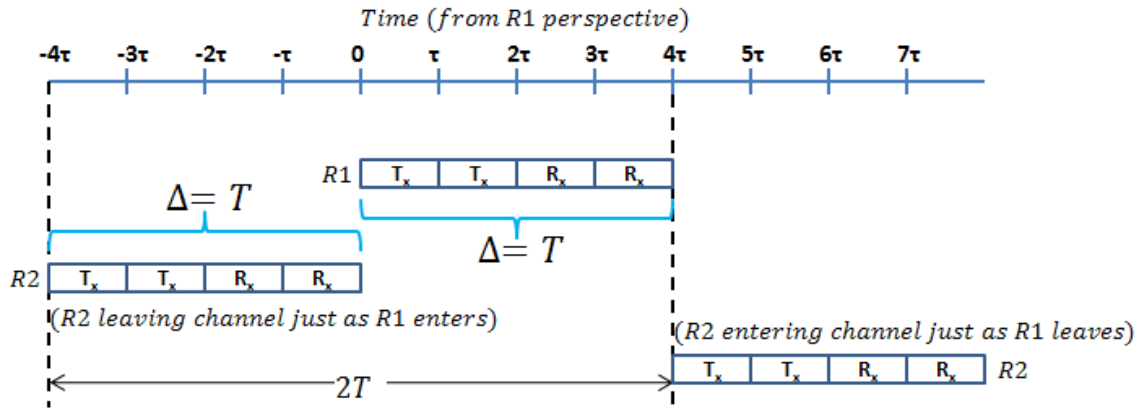

Figure 12: Total time in which Receiver and Transmitter will be in same channel for some period of time; here each radio is in the same channel
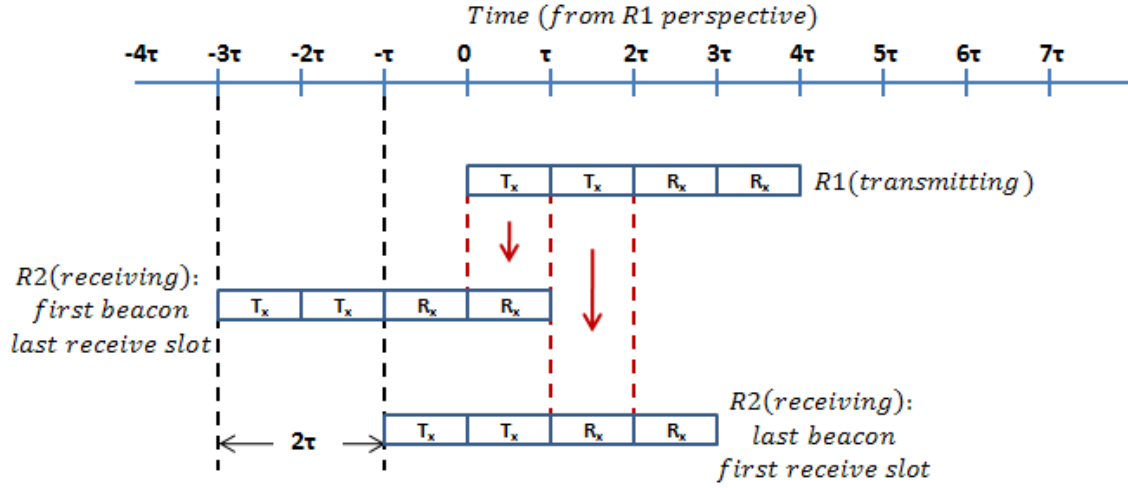
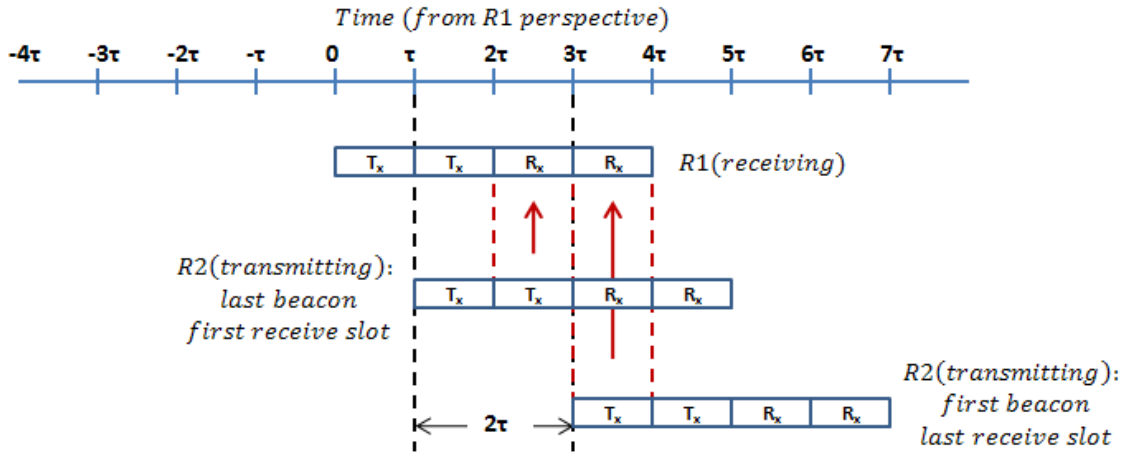*Figure 13: R2 receive time range, both radios in same channel*

Figure 14: R1 receive time range, both radios in same channel

4.1.2 Simple Case with Propagation Delay

This simple analysis does not take propagation delay into account. Let d be the propagation delay between two radios. In most cases, propagation delay will be negligible compared to the time slot length. However, in some cases, depending on the amount of propagation delay, the possible beacon receive windows for each radio may overlap or completely miss each other altogether. We assume that the propagation delay

will not be so great that the transmitting radio will have left the channel before a response packet arrives. The overlap of possible beacon times is still an issue because it effectively reduces the total *unique* receive time. This means when calculating the probability of receiving a beacon, we must subtract the *overlap* out of total receive time so that it does not get counted twice. Figure 15 and Figure 16 show this overlap for $R2$'s and $R1$ receive windows respectively. It may look as though the total receive time is still 4τ; however, if we look at the receive time ranges plotted on a number line (Figure 17), we see how they end up overlapping. Thus if propagation delay causes any overlap in receive windows, the probability of meeting in the same channel is further reduced.
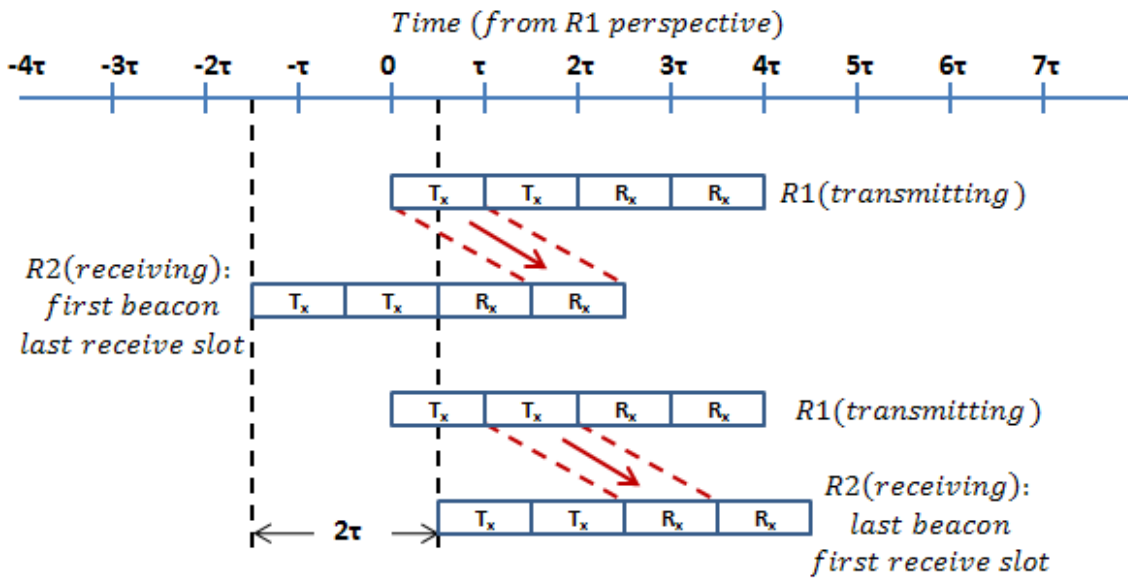

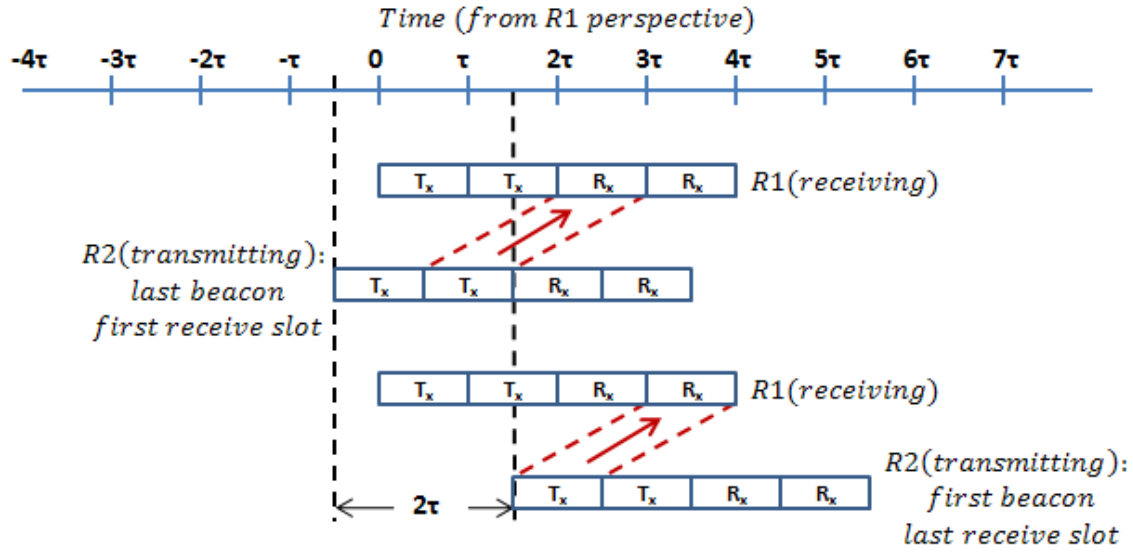*Figure 15: R2 receive range with delay, both radios in same channel*
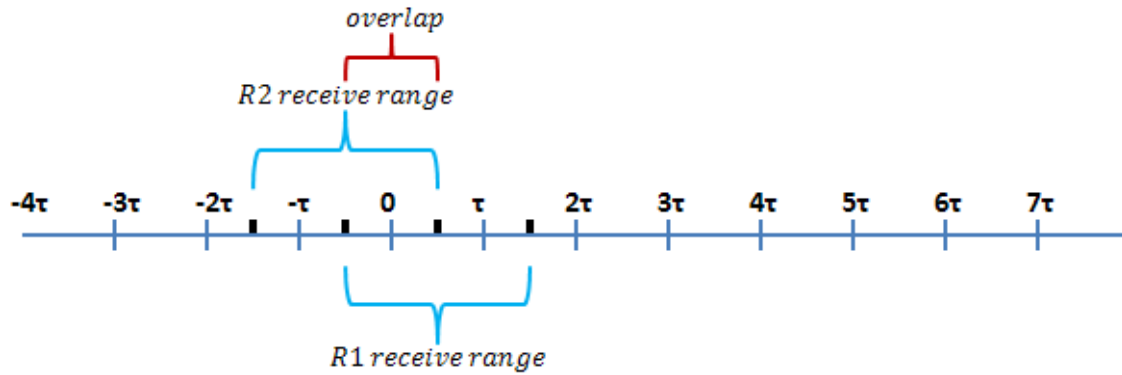
Figure 16: R1 receive range with delay

Figure 17: Receive times overlaid on number line

While too much propagation delay can be detrimental to the handshake process (radios leave the channel before beacons/responses can arrive), smaller delays can actually be beneficial. This is primarily due to the 2τ failure window discussed in Section 3.1.2. For example, Figure 18 shows the alignment of a failed beacon reception. In this instance, both radios will be unable to receive each other's beacons. However, Figure 19 shows the same alignment (with respect to R2), but with propagation delay.

Here, we can see that even though R1 cannot successful transmit a beacon to R2, R2 is able to transmit a beacon to R1.
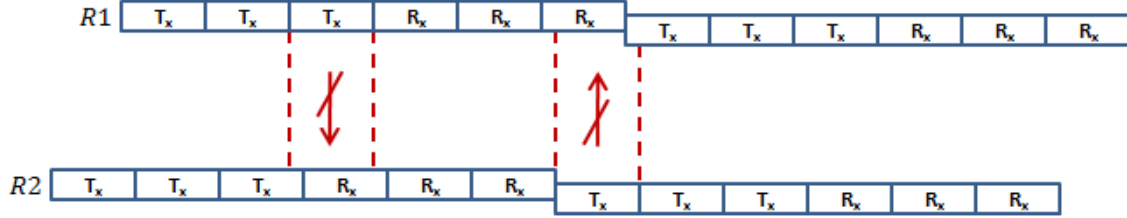

Figure 18: Failure Case with no Delay, both radios in same channel
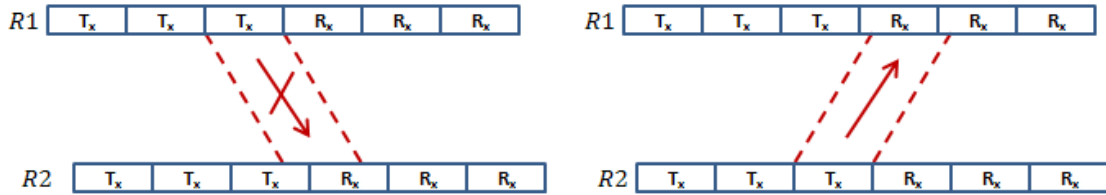

Figure 19: Failure Case one-way, successful the other way, both radios in same channel

The propagation delay actually *reduces* the failure window for the handshake until it is gone. However, after that point propagation delay reduces the total time that a successful beacon can be received in proportion to the total time in the channel, leading to worse performance with too much delay. We will return to this effect in the next section.

4.1.3 General Derivation of Probability of Initial Beacon Reception

We now expand these observations to create general analytical equations for the probability of a radio receiving an initial beacon from another radio given they are in the same channel for some period of time. These will later be expanded to include the reply transmissions to complete the connection. The general equation for the probability is

The total time in the same channel is always double the sum of transmit and receive slots used. Using the syntax discussed earlier, we denote this as $2(m + n)$. To calculate the numerator, we know from Section 3.1.2 that without propagation delay each radio has a 2τ failure window (4τ total) resulting in $2(m + n) - 4\tau$ possible receive time.

$$Receive\ range = 2(m + n) - 4\tau = 2(m + n - 2\tau) \qquad (7)$$

Another way to derive this is to examine how the receive range is calculated. Since a node requires a full transmit to determine any valuable information, the earliest and latest a node can receive is right after the first receive slot has entered the same time range as the transmit and right before the last receive slot has left the transmit range. The range between these two times is $n - 2\tau$. We can then add the number of receive slots used to get $m + n - 2\tau$. Figure 20 illustrates this for the case without propagation delay. We must double this value to take into account both nodes.
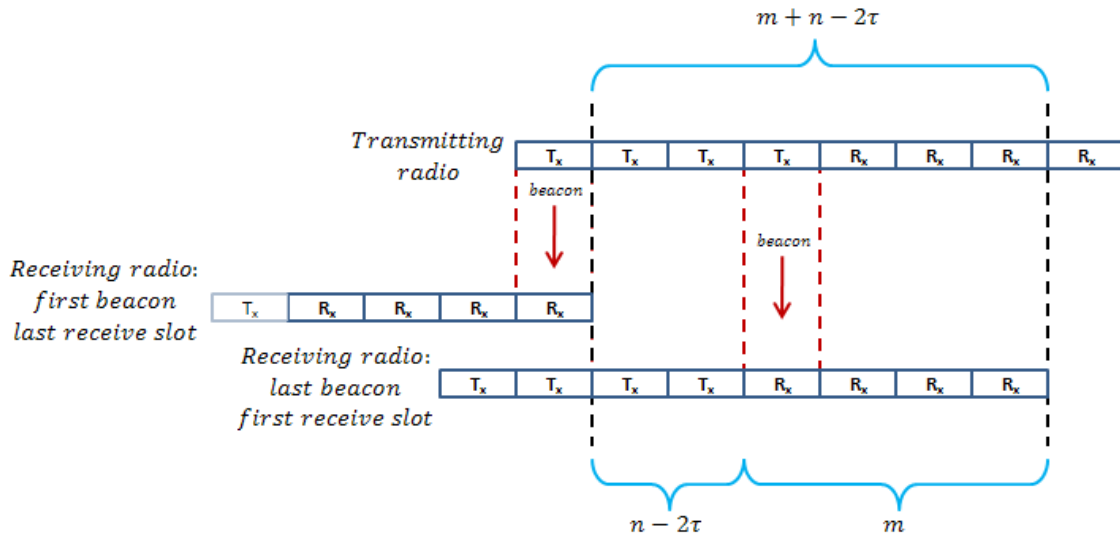

Figure 20: Total node receive range, both radios in same channel

To determine the probability of a successful beacon reception without propagation delay, we take the total possible range of receive time, divided by the total time in the channel:

$$P(receive\ beacon|\ d = 0) = \frac{2(m + n)\tau - 4\tau}{2(m + n)\tau}$$

Now we extend this to deal with propagation delay. This means removing overlap from the beacon receive time.

$$P(receive\ beacon) = \frac{2(m + n - 2\tau) - overlap}{2(m + n)}$$

To define overlap, we look back to Section 4.1.3. Since propagation delay affects both nodes by causing their receive windows to effectively shift towards each other, the overlap caused by a propagation delay d would be $2d$. However, since we have that 2τ failure window, as long as $2d \leq 2\tau$, the propagation delay will shrink the failure window, increasing the probability of beacon reception. When $2d > 2\tau$, the initial failure window is gone, but the propagation delay will begin causing the overlap in receive times, reducing the probability. Therefore we have two equations for these scenarios.

$$P(receive|d \leq \tau) = \frac{2((m+n-2)\tau+2d)}{2(m+n)\tau} = \frac{(m+n-2)\tau+2d}{(m+n)\tau} \qquad (8)$$

$$P(receive|d > \tau) = \frac{2(m+n)\tau-(2d-2\tau)}{2(m+n)\tau} = \frac{(m+n+1)\tau-d}{(m+n)\tau} \qquad (9)$$

Figure 21 is a graphical representation of the probability of initial beacon reception with four transmit and receive slots. The graph peaks at $12.8\ \mu s$, which is τ, the turning point between the two equations, with a 100% probability of reception.
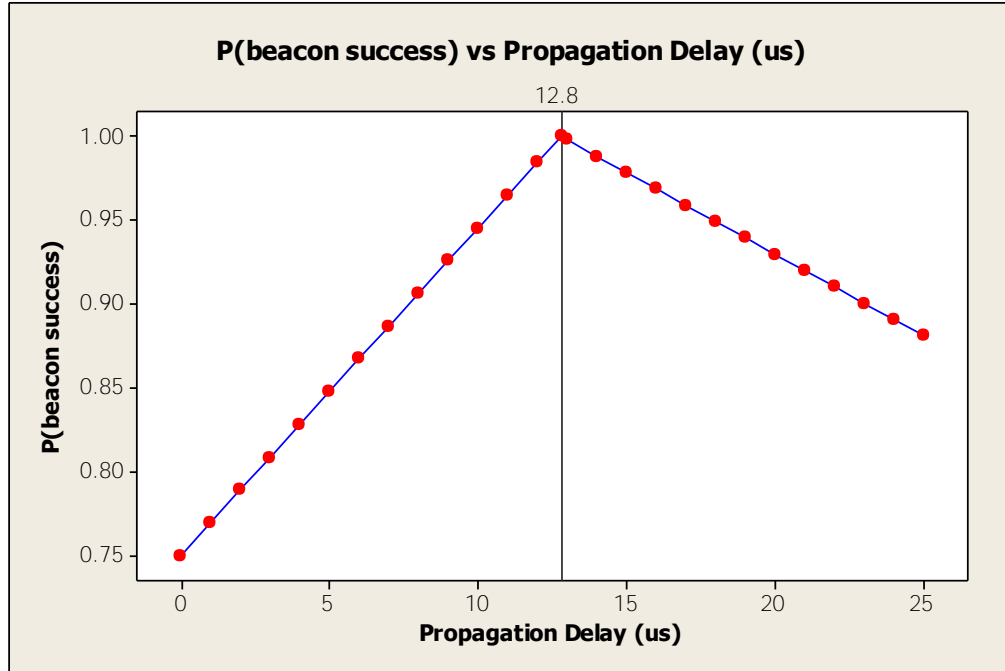
Figure 21: Probability of initial beacon reception vs. propagation delay with n=4, m=4

4.1.4 Probability of Response Packet Reception

Now that we have determined the probability of two radios receiving the other's beacon, we look at the probability of receiving the response to the beacon. We will examine the problem both without propagation delay and then with. In our handshake implementation, we assume that as soon as a full beacon packet is received, a radio will immediately respond. Within the window of time in which two radios could receive a beacon from the other, we look at the extremes - the first and last times a beacon could be received. From here on, the transmitting radio is characterized by the radio whose initial beacon is successfully received. The receiving radio is characterized by the radio that receives the initial beacon and sends a response.

We examine a response when the receiving radio receives the transmitting radio's first beacon packet during its last receive slot, and when the receiving radio receives the

transmitting radio's last beacon during its first receive slot. Figure 22 and Figure 23

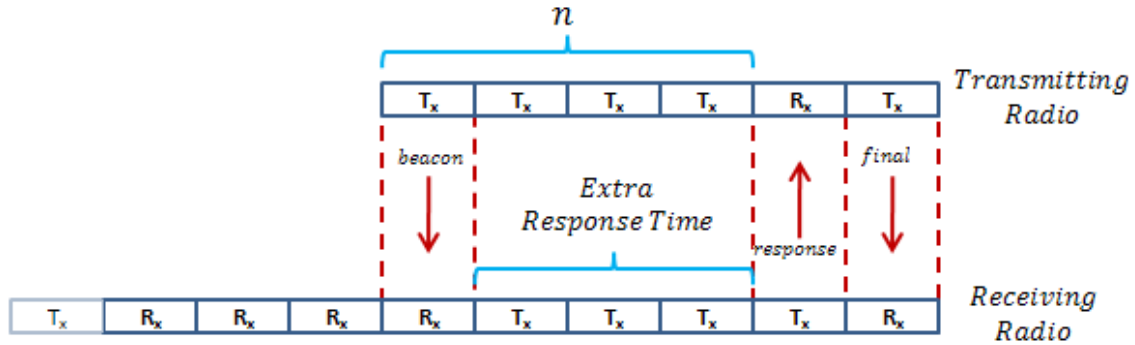detail the two scenarios without delay and Figure 24 and Figure 25 with delay.

Figure 22: No propagation delay, first transmit, last receive slot, both radios in same channel
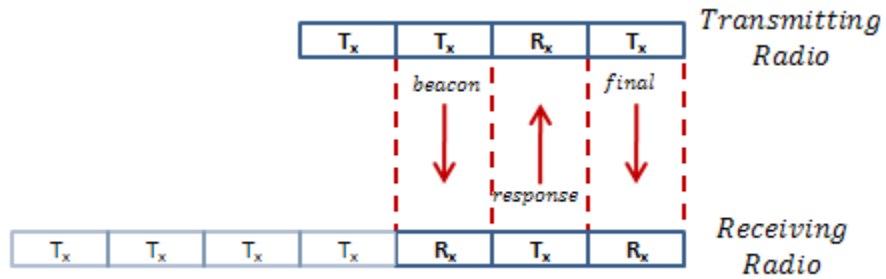

Figure 23: No propagation delay, last transmit, first receive slot, both radios in same channel
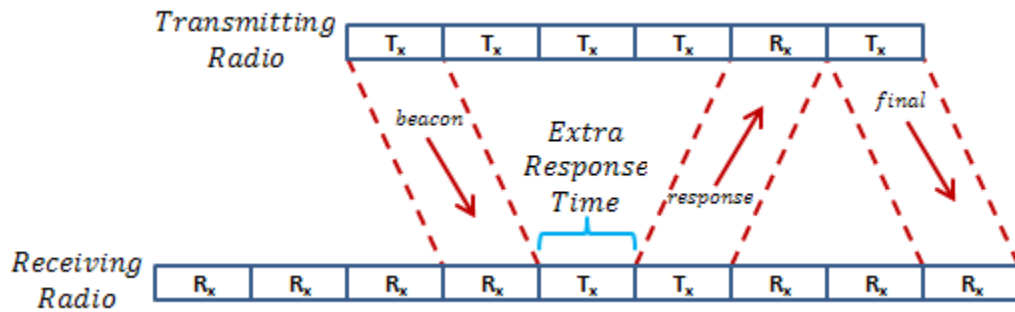

Figure 24: Propagation Delay, first transmit, last receive slot, both radios in same channel
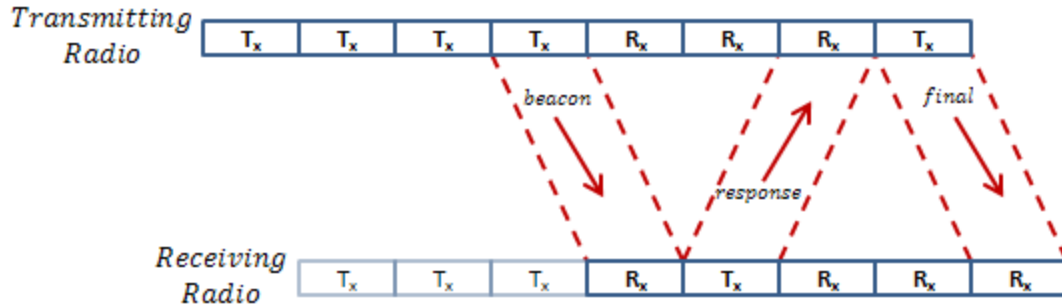
Figure 25: Propagation Delay, last transmit, first receive slot, both radios in same channel

Since the receiving radio begins transmitting immediately following reception of a beacon, the case when the first beacon is received in the last time slot would require the receiving radio to transmit its response n times to guarantee the transmitting radio receives it. In the second case, the receiving radio would need to send only one response to be heard. However, without any extra information being sent, the receiving radio would not know which case it is in, which makes it impossible to know how many responses would be necessary to be heard. If the receiving radio tried to send the minimum number of responses each time, the only successful responses would be those responding to the last beacon. Therefore, to guarantee a response being heard, it would have to use the maximum number of response transmissions each time. The same scenario applies to the final response beacon. Because the maximum number of transmits must be used to send the response, the effect is reciprocated to the transmitting radio sending its final response. Therefore, the transmitting radio must spend n time slots transmitting the final response.

One solution to this problem is to add information to the beacon, for example in a header or in the modulation of the beacon. Since the radios are using symmetrical rendezvous, each radio can use this information in conjunction with the m and n values

to intelligently respond to the beacon. For example, if there are $n = 5$ total transmissions, the second beacon would have information indicating it was the second beacon. The receiving radio would know how many transmissions are remaining, and would know when the transmitting radio would be able to hear a response. Therefore, instead of transmitting the maximum of $n = 5$ responses, the receiving node could save power by waiting until it knows the transmitting radio is finished transmitting, and then send the response.

Propagation delay also plays a factor in how soon the receiving radio can send its response. Assuming the radios have a means with which to calculate propagation delay, the receiving radio would know when to send the response so that, accounting for the delay, the transmitting radio receives the response as early as possible. This allows radios to send the response d time units earlier than without delay. However, it is unlikely that the radios will be able to calculate the propagation delay before rendezvous occurs.

4.1.6 Backoff Delay due to Timeout

When two radios are offset in such a way that they will be unable to successfully handshake (ie. they are in one of the failure windows discussed in Section XX), all subsequent handshakes will fail until this offset is fixed. In order to remove this degenerate case, we implement a delay into the handshake algorithm should the rendezvous take longer than expected. To guarantee that the handshake will work in the next iteration, the delay implemented at one of the radios must be at least 2τ so that they will move out of the failure range.

However, because we are using symmetric rendezvous, we cannot have just one of the radios implement the delay. Since both radios must have a delay, there is a chance

that they could both delay by a complimentary amount, thus returning to the failure

range. Therefore, we need to ensure that the net delay implemented by both radios with

respect to each other is at least 2τ. For example, if one radio delays for τ, the other must

delay at least 3τ to guarantee being successful.

To solve this problem we look to randomly choose the delay from a uniform

distribution of delays for each radio, from 0 up to b timeslots of delay, where $b \geq 2$.

We also know that the possible failure offset windows are between $-\tau$ to τ, $-(m+n)\tau$ to

$-(m+n-1)\tau$, and $(m+n-1)\tau$ to $(m+n)\tau$. Let R be the uniform probability

distribution of having any of these possible failure offsets, D be the uniform distribution

of the delay range (from 0 to b), and R' be the new offset distribution after adding the

delays. If we are implementing a delay, know we are operating under distribution R. R'

is equal to the difference between the two radios' delays plus the original offset.

$$R' = D_{radio2} - D_{radio1} + R.$$

We want to know the probability that R' is also in the failure offset given the radio

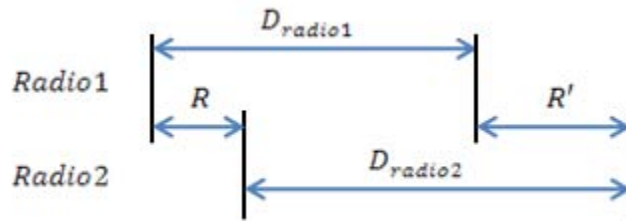delays. Figure 26 gives a graphical representation.

Figure 26: Offset range based on Radio Backoff Delay

Because R, D_{radio1}, and D_{radio2} are independent random variables, we must perform a sum of independent random variables to determine the probability that R' falls within the failure range. We begin with the pdfs of both D and R ($D_{radio1} = D_{radio2}$) in Figure 27.

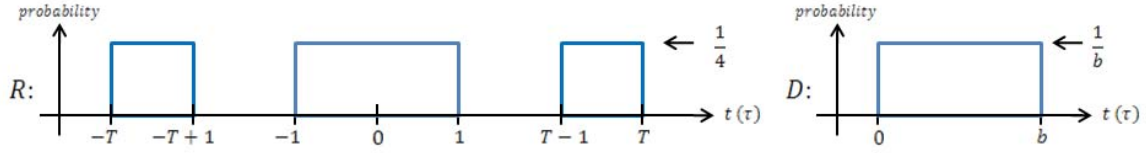*Figure 27: Probability Distributions of R and D*

By performing a convolution of R with D, a new pdf is created ($D_{radio2} + R$ part of the equation). As D is convolved with R, each block on R will form a separate block in $R+D$. However, if b is sufficiently large, these new blocks will begin to merge. Figure 28 shows two different resulting pdfs created in Matlab. In the same way, when $R+D$ is convolved with the negative pdf of D (-D_{radio1} part of the equation), the blocks will begin to merge together. Figure 29 shows two different final pdfs for R'.

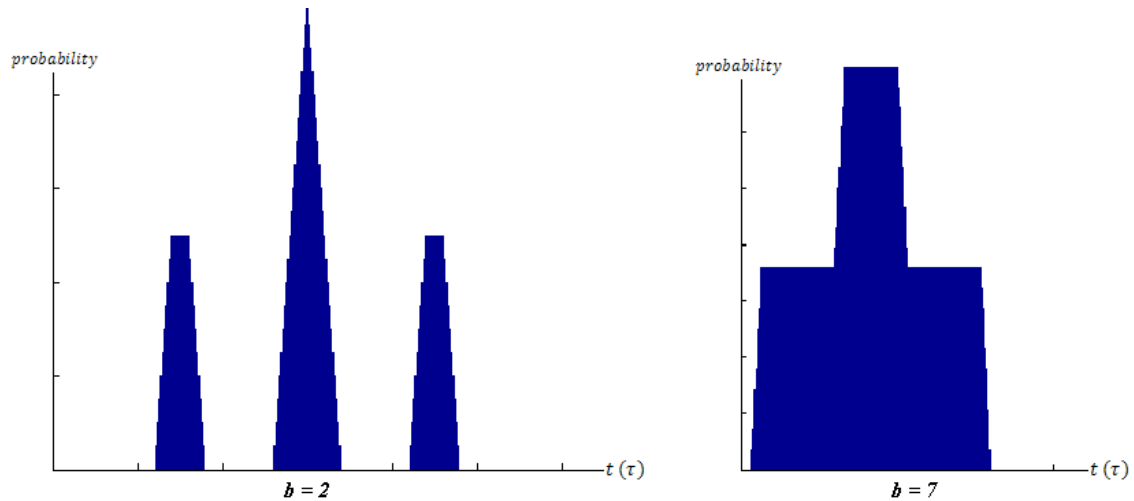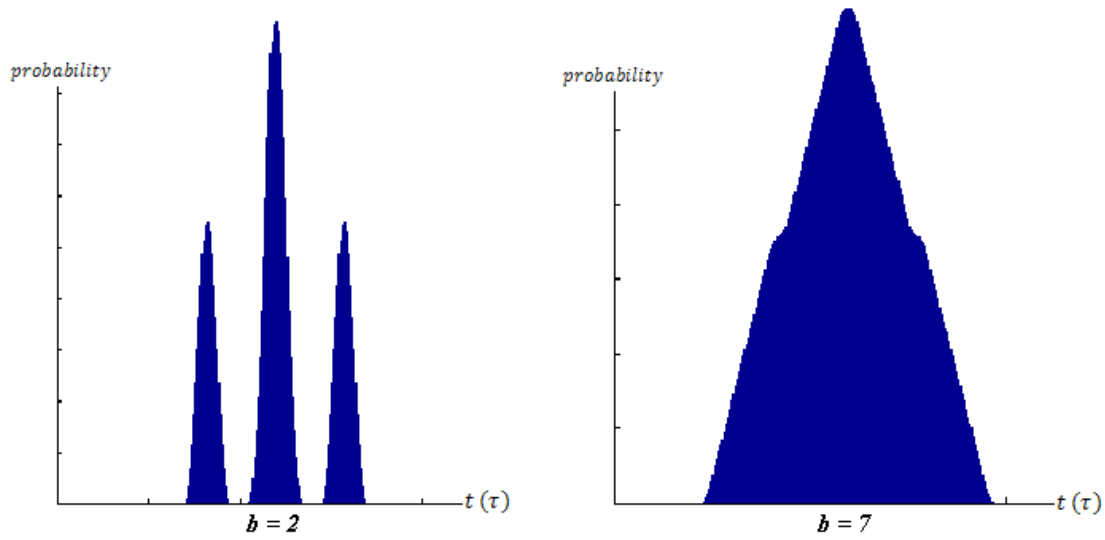*Figure 28: Estimated Probability Distribution for R + D, with b = 2 and b = 7*

Figure 29: Estimated Probability Distribution for R', with b = 2 and b = 7

By analyzing the values in the failure ranges on *R'*, we can determine the probability that

the handshake will fail again. Figure 30 shows the resulting probabilities of recurring

failure for given backoff ranges. This plot is also independent of *m* and *n*.

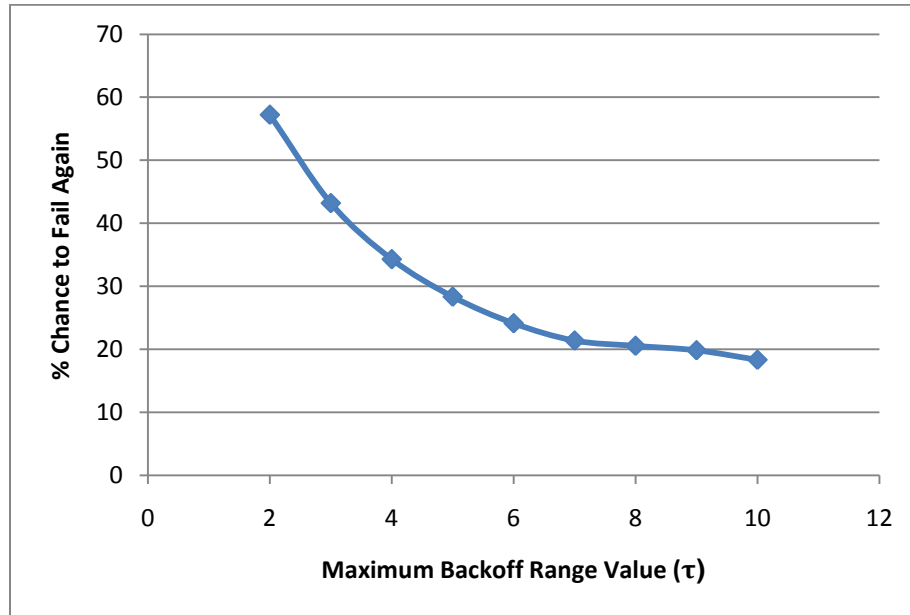*Figure 30: % Chance for handshake to fail again given certain backoff range values; holds for any configuration of n and m*

As b increases, the probability of failure decreases; however, b also increases the amount of time spent in a particular channel. In our implementation, a radio will listen during this delayed time, which may increase the chances of being discovered by another radio or a jamming radio. If a radio is made idle during the delay time, the increased delay would likely hurt the rendezvous process more than the decreased probability of handshake failure would help.

4.1.7 Time to Handshake (TTH)

The expected TTH is different for both the receiving and transmitting radio. Because the receiving radio enters the channel earlier and receives the final beacon to complete the handshake, the total time spent handshaking is calculated relative to the time spent by the receiving radio. The expected time for the receiving radio is a function of the number of transmit slots n, the number of receiving slots before it receives a beacon, the slot spent receiving the beacon, the wait time until it is known that the transmitting radio is receiving, the slot spent transmitting the response, and the time spent waiting for a response back. Of these, the number of transmit slots, n, and the slots spent receiving the beacon, responding to it, and receiving the final beacon are constant for a total of $(n + 3)\tau$. The variable values are the *receive time before the beacon arrives* and the *extra response time before sending an effective response*. These are combined into a general equation for TTH.

$$TTH = (n + 3)\tau + t_{before\ beacon\ received} + t_{extra\ response}$$

The following lemmas and proposition define these variables in terms of $n, m,$ and Δ, and give a complete equation for the TTH.

We begin by first examining two lemmas.

Lemma 1. $t_{before\ beacon\ received} = \max\{0, \Delta - n\tau\}$

The time before a beacon can be received can range from 0 (the beacon is heard immediately upon receiving in the channel) up to $(m - 1)\tau$ (the beacon is heard during the last possible receiving slot). Figure 23 and Figure 22 illustrate these extremes, respectively. The first time in which the receiving radio can receive is the number of transmitting slots subtracted from Δ. For example, if the receiving radio entered the channel at time -5τ with respect to the transmitting radio (who starts at time 0τ), with $n = 3$, the first receiving slot would start at time -2τ. Since the beacon cannot be received before time 0τ, there is a 2τ gap of receiving slots that will not hear anything. Therefore, the time spent receiving before hearing a beacon is the $\Delta - n\tau$. However, because this receiving time cannot be a negative value, we must incorporate a maximizing function.

Next, we look at the waiting time between receiving a beacon and sending the response.

Lemma 2. $t_{extra\ response} = \max\{0, \min\{(n - 1)\tau, \Delta - \tau\}\}$

Depending on which beacon is heard, the receiving radio must wait from 0 time (if the last beacon is heard) up to $(n - 1)\tau$ time (if the first beacon is heard). Which beacon is heard is again dependent upon Δ. Without factoring in propagation delay, the lowest value of Δ when the receiving radio can hear a beacon is τ. Therefore, when Δ is at τ, the wait time is 0. By subtracting τ from Δ we can get the wait time. However, if we simply subtract τ from Δ for the upper bound, we can go outside the maximum wait time of $(n - 1)\tau$. Using the example earlier, if the receiving radio entered the channel at time -5τ, Δ would be 5τ. Subtracting τ from this gives us a waiting time of 4τ, which

would mean that the transmitting radio is transmitting four beacons, which is obviously not the case. Therefore we place the upper bound of the wait time at $(n - 1)\tau$. In the same way, if the two radios are exactly aligned, with $\Delta = 0$, the response time would be $-\tau$. Since the response time cannot be negative, we place a maximizing function on it. By combining these equations, we receive the TTH.

Proposition 1. $TTH = (n + 3)\tau + max\ \{0, \Delta - n\tau\} + \max\{0, \min\{(n - 1)\tau, \Delta - \tau\}\}$

4.1.8 Effects of Propagation Delay on TTH

With no propagation delay, it is easy to assume that radios can predict when to transmit and receive based on message information and symmetric rendezvous. It is also likely that many of these radios will operate in close enough proximity to each other where propagation delay is negligible. However, in the cases where propagation delay cannot be ignored, it is unreasonable to assume that radios can estimate propagation delays between radios they have not met yet. At the very least, using some sort of time stamp, it would require a beacon and response before a radio could calculate propagation delay, at which point rendezvous would have occurred anyway.

Propagation delay affects the TTH by changing the effect that Δ has on the handshake process. When the delay increases, the receiving radio needs to enter the channel later relative to the transmitting radio in order to receive the initial beacon. This effectively reduces Δ by d, the propagation delay experienced. This means that the equation for the TTH is affected in three terms: the receive time before the beacon is heard, the time before an effective response is sent, and the receive time spent after a response is sent. The receiving radio will still transmit for n time slots, as well as spend a time slot to receive the beacon and send a response. However, since it will take d time

for the response to arrive and d time for the final response to be received, the time spent receiving the final response is increased by $2d$. Therefore, the constant term becomes $n\tau + 3\tau + 2d$. Next, we must figure out how the rest of the terms are affected by the propagation delay.

Again we look at the following three lemmas.

Lemma 3. $t_{before\ beacon\ received} = \max\{0, \Delta - n\tau + d\}$

The receive time before hearing the beacon is still a function of the Δ and n. The time spent receiving before transmitting a beacon is also independent of the delay. Therefore, since Δ has been reduced by d, we must add it back in to get the equivalent receive time without delay. Figure 31 shows the new time values associated with Δ, n, and d compared against each other to form the time spent receiving.
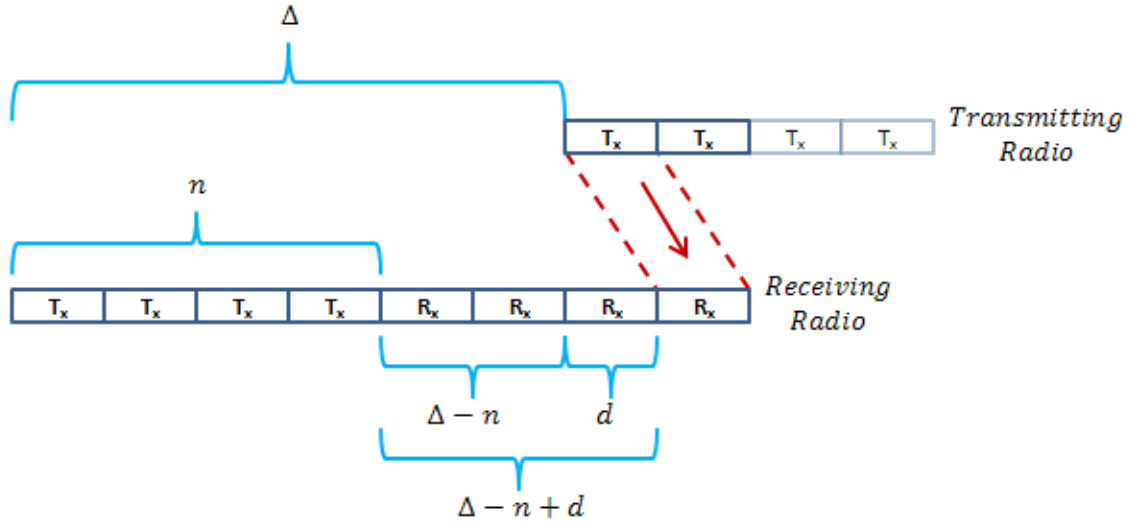

Figure 31: Receive time before beacon heard as a function of Δ, n, and d, both radios in same channel

Lemma 4. $t_{extra\ response} = \max\{0, \min\{(n-1)\tau, \Delta - \tau\}\}$

66

The response time is affected by propagation delay both coming from the transmitting radio, and going out from the receiving radio. Relative to the transmitting radio, the time at which the receiving radio receives the beacon and can send a response are later and earlier, respectively. However, since we assume that propagation delay is unknown, the radios must act as though there is no propagation delay and thus this term remains unchanged. As an example, if a radio overestimated the propagation delay and sent a single response assuming it would arrive after the other radio finished transmitting, the response would collide and no handshake would occur. By sending responses as if there were no propagation delay, the handshake may take longer, but does not run the risk of trying to estimate the propagation delay. Figure 32 shows the response time when propagation delay is introduced and the first beacon is received during the last receive slot. Comparing with Figure 22, the response time remains unchanged. However, since the propagation delay causes the transmitting radio to receive the response later, the final response is sent later, resulting in a longer handshake.
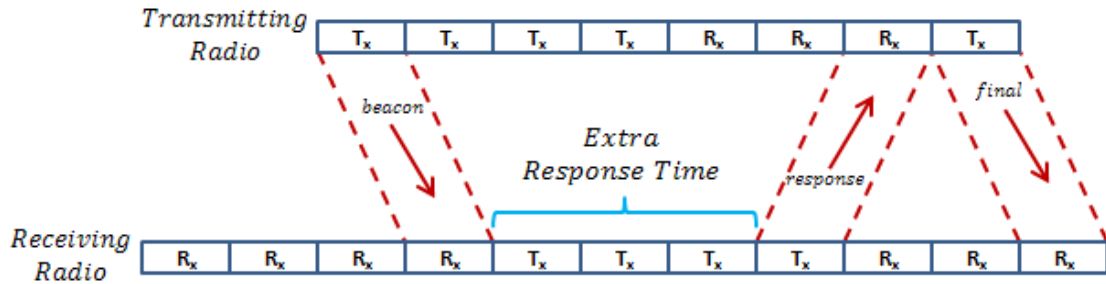

Figure 32: Response time remains unchanged with delay, both radios in same channel

These equations are finally combined into the new equation for TTH.

Proposition 2. $TTH = (n + 3)\tau + 2d + \max\{0, \Delta - n\tau + d\} +$

$$+\max\{0, \min\{(n - 1)\tau, \Delta - \tau\}\}$$

We now wish to find the expected value of TTH. We being by looking at the maximum and minimum values of Proposition 2.

Lemma 5. $TTH_{min} = (n + 3)\tau + 2d$

Lemma 6. $TTH_{max} = (n + 3)\tau + \Delta - n\tau + d + n\tau - \tau + 2d = (2n + m + 1)\tau + 3d.$

Due to the maximizing functions, the absolute minimum value for the TTH would be $(n + 3)\tau + 2d$ (both maximizing functions yielding 0). To figure out the maximum value of this function, we must look at the variables containing Δ since in this case, it is the only varying parameter. The output of the first maximizing function should be the second term, $\Delta - n + d$, since 0 is the minimum output. Looking at the maximin function, to get the highest value for TTH, we would like the minimum function to be as large as possible. Since the first term does not vary, it cannot get smaller, so therefore we want the output of the minimum function to be $(n - 1)\tau$. In order for this to be true, the second term must be greater than or equal to it.

$$\Delta - \tau > (n - 1)\tau \quad \text{or} \quad \Delta > n\tau$$

In order for the maximin function to produce its largest output, the offset Δ must be greater than $n\tau$. Therefore, the largest value for TTH would have to be a function of the static parameters, the second term of the first maximizing function, and the first term of the minimizing function in the second maximizing function. From Section 3.1.2 we know that each radio has a failure range from $(T - 1)\tau$ to T. Therefore, a Δ that results in a successful handshake has an upper bound of $(T - 1)\tau = (n + m - 1)\tau$, and thus, the maximum value of $TTH_{max} = (2n + m + 1)\tau + 3d.$ To determine the expected value of the TTH, we graph the TTH over the range of possible values for Δ in Figure 33.

By taking the area under the line, and dividing by the total range of values, we can determine the expected TTH, given the handshake is successful.
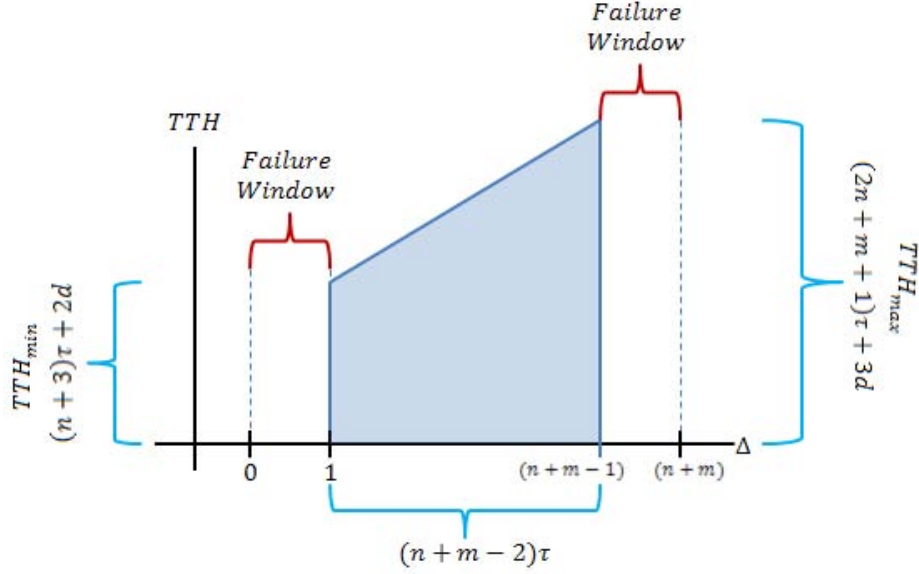

Figure 33: Graphical Representation of TTH vs. Δ to determine E[TTH]

$$Area = \left\| [(n+3)\tau + 2d] + \frac{1}{2}[(n+m-2)\tau + d] \right\| * (n+m-2)\tau$$

$$E[TTH] = \frac{Area}{(n+m-1)\tau} = (n+3)\tau + \frac{1}{2}(n+m-2)\tau + \frac{5}{2}d \qquad (10)$$

For example, if a radio was using four transmit slots and four receive slots, assuming no propagation delay, we would expect a total handshake time of $0.128\ ms$.

$$E[TTH|n=4, m=4, d=0, \tau = .0128\ ms] = 7\tau + 3\tau + 0 = 0.128\ ms$$

4.2 Simulation Analysis of Handshake

All simulations were run using the default values from Table 2, unless otherwise stated.

4.2.1 Effects of the Handshake

The first simulations were run to test the handshake and compare to the analytical results obtained in the previous section. We first examine the increase in TTR (both raw

and percentile) for various time slot setups compared to the TTR without a handshake (TTM). Next we examine probability of a successful beacon compared with the simulated results for the probability of a successful handshake. Then we look at the number of steps (channel changes) needed before rendezvous. Finally, we compare each of the actual TTR values for the handshake.
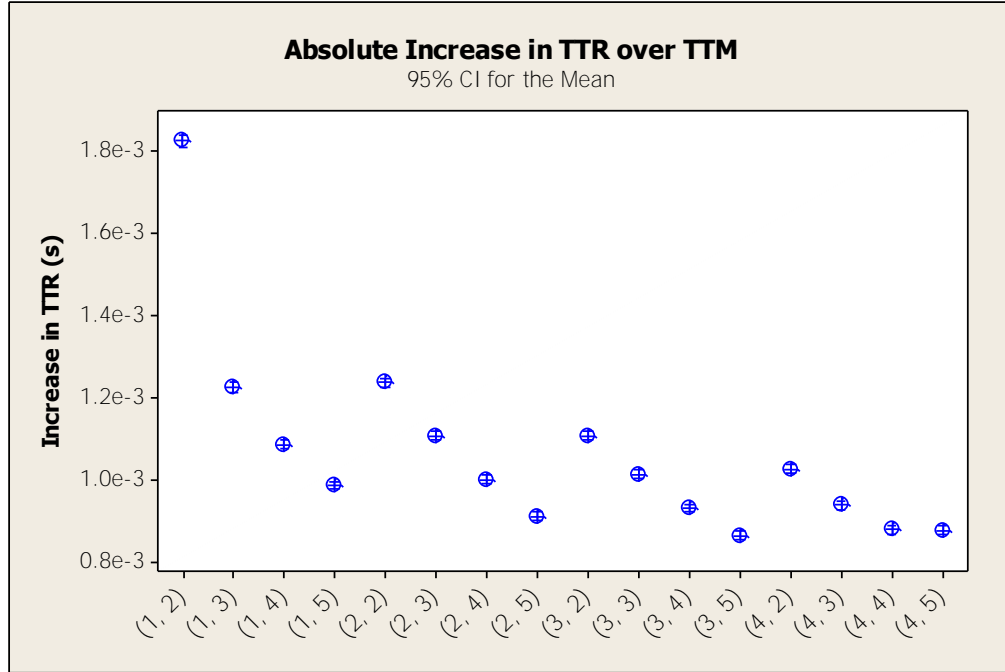

Figure 34: Increase in TTR due to handshake for various transmit and receive slots (n, m)

Figure 34 and Figure 35 shows that as the number of transmit and receive slots increase, both the absolute and percentage increase in TTR from TTM decreases. An increase in any single slot count (for either transmit or receive) will decrease the absolute and percentage increase in TTR over TTM, as indicated by the steeper slopes on the left side of the graph.
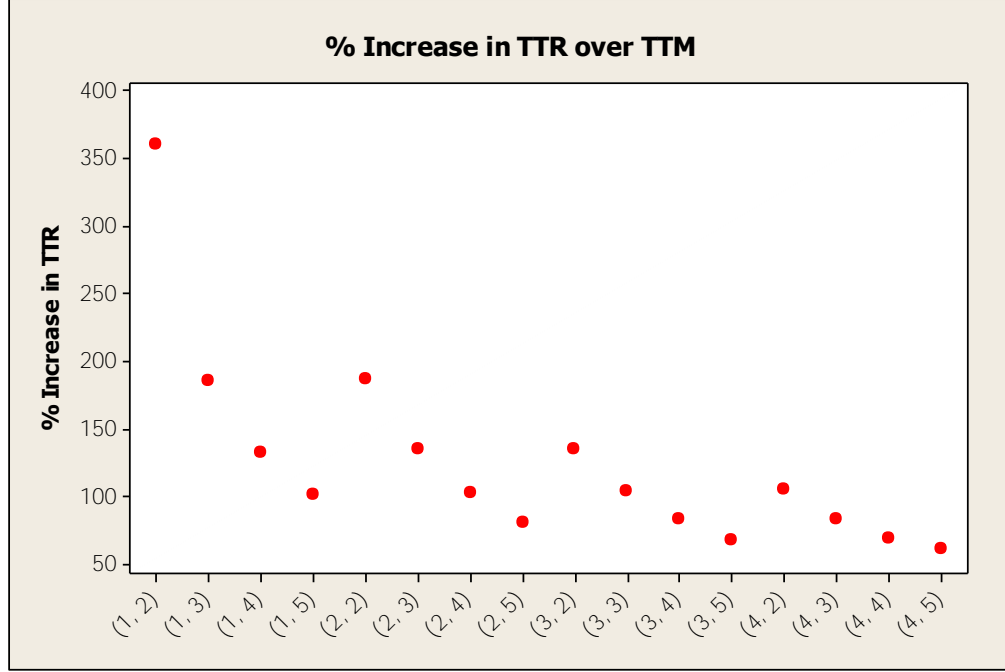
Figure 35: Percentage Increase in TTR due to handshake for various transmit and receive slots (n, m)

Figure 36 and Figure 37 show the probability of successful beacon reception and a successful handshake respectively for various transmit and receive slots. Here we see that the probability of a successful handshake increases as the number of transmit and receive slots increase. It should be noted that the probability is a function of total time slots used, which is why several combinations yield very similar results (i.e., 1n, 4m ≈ 2n, 3m).

The simulated results are significantly lower than those calculated, with about 10% lower probabilities. This is due to the fact that during simulation, the handshake failures are not always independent. When described in Section 4.1.4, the probability of a beacon reception is only for a single handshake instance. For example, if two radios attempt to handshake, they have the same probability of handshaking regardless of when the handshake takes place. However, if these handshakes take place before the backoff

delay is implemented, the failures become correlated, because the first handshake failure would mean any subsequent handshakes before backoff would fail. This results in an overall lower probability of handshaking. The simulation takes this correlation into account when calculating the probabilities, which is why those values are lower.
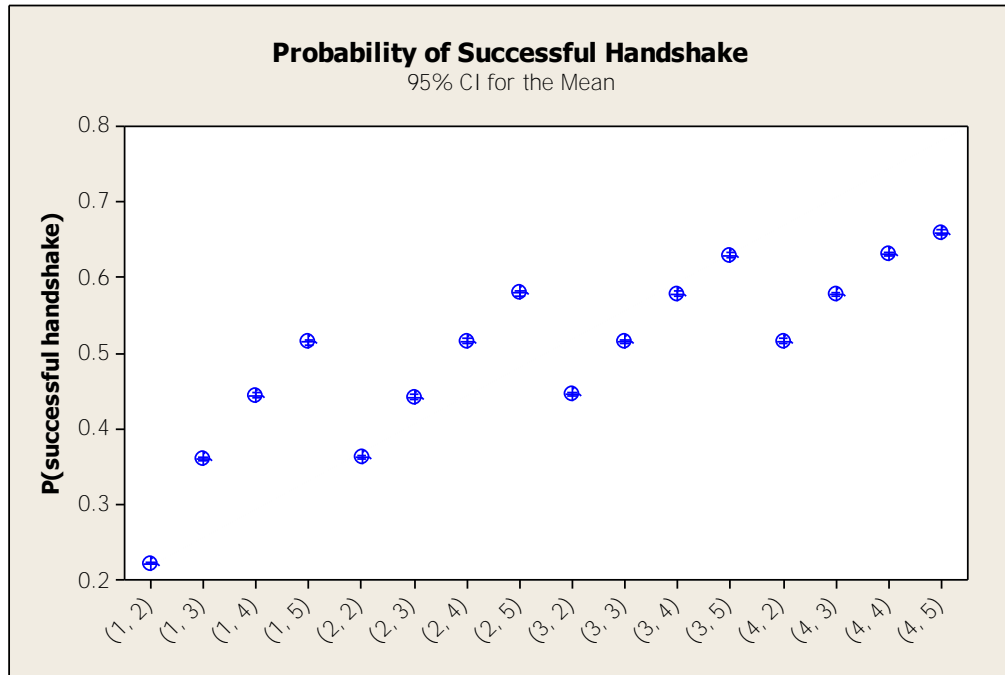

Figure 36: Probability of a successful handshake given certain transmit and receive slots (n, m); these results were derived from the simulation
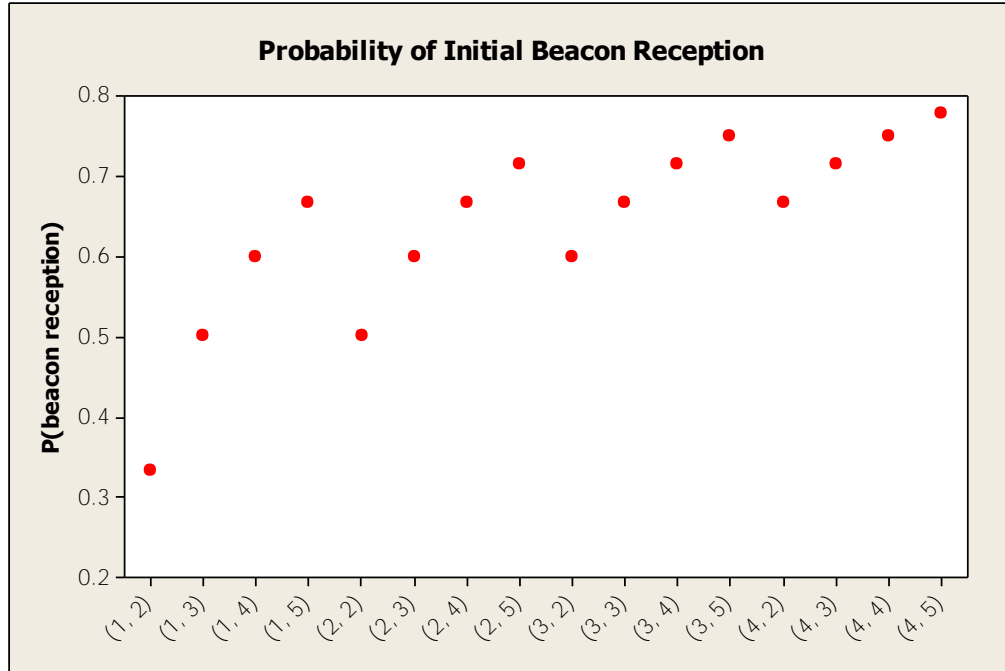
Figure 37: Probability of successful initial beacon reception given certain transmit and receive slots (n, m); these results are derived from Equations 7 & 8
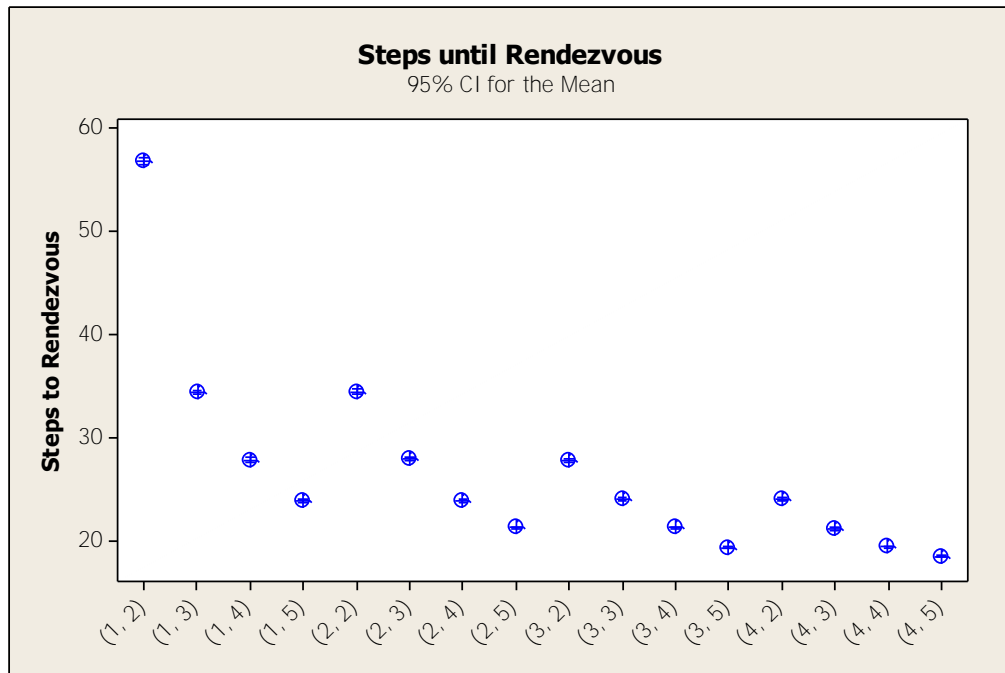

Figure 38: Steps until rendezvous for various transmit and receive slots (n, m) where a step is equivalent to a channel change

73

Figure 38 shows that as the number of transmit and receive slots increase the number of steps necessary to rendezvous decrease. This is also due to the fact that the higher number of transmit and receive slots increase the probability of a successful handshake. A step is equivalent to a channel change, and each radio is in a channel for T time slots.
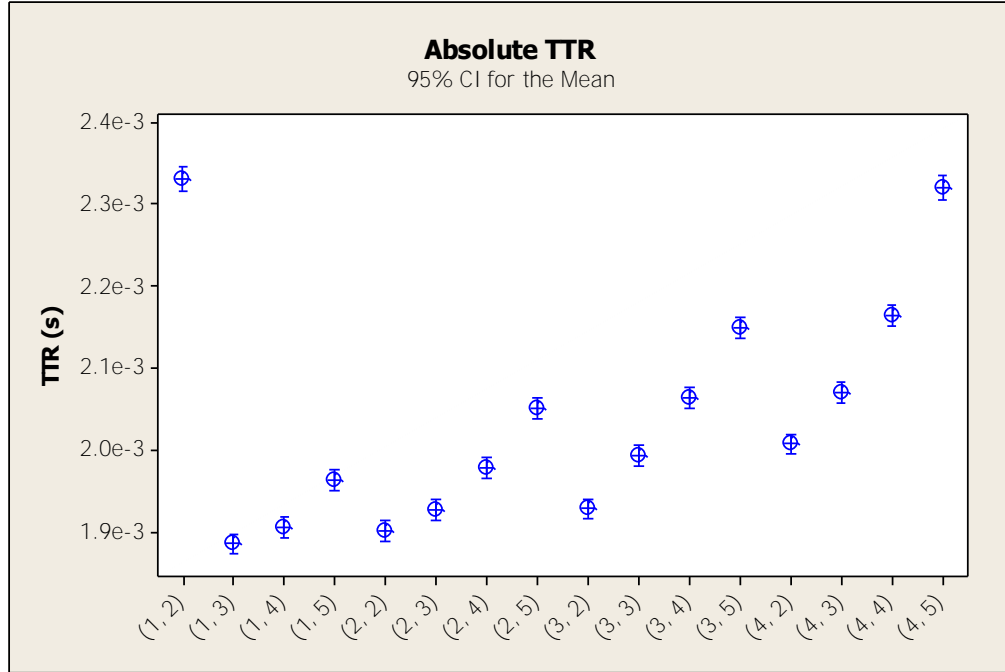

Figure 39: TTR for various transmit and receive slots (n, m)

Figure 39 may seem contradictory to Figure 35; however, this is actual TTR, not the increase. Despite that the higher transmit and receive slots increase the probability of a successful handshake, and even reduce the number of steps necessary to rendezvous, it does not make up for the time spent in each channel attempting to handshake. In other words, the longer handshaking process overwhelms any gains from having a more successful handshake.

4.2.2 TTH Analysis vs. Simulation

Next we look at the expected TTH during the simulations and compare it to the E[TTH] calculated earlier in Section 4.1.8. Figure 40 shows the E[TTH] for both the equation and simulation. The small intercept ($8\text{x}10^{-7}$) relative to the axis values indicates that the values for both analysis and simulation are similar.
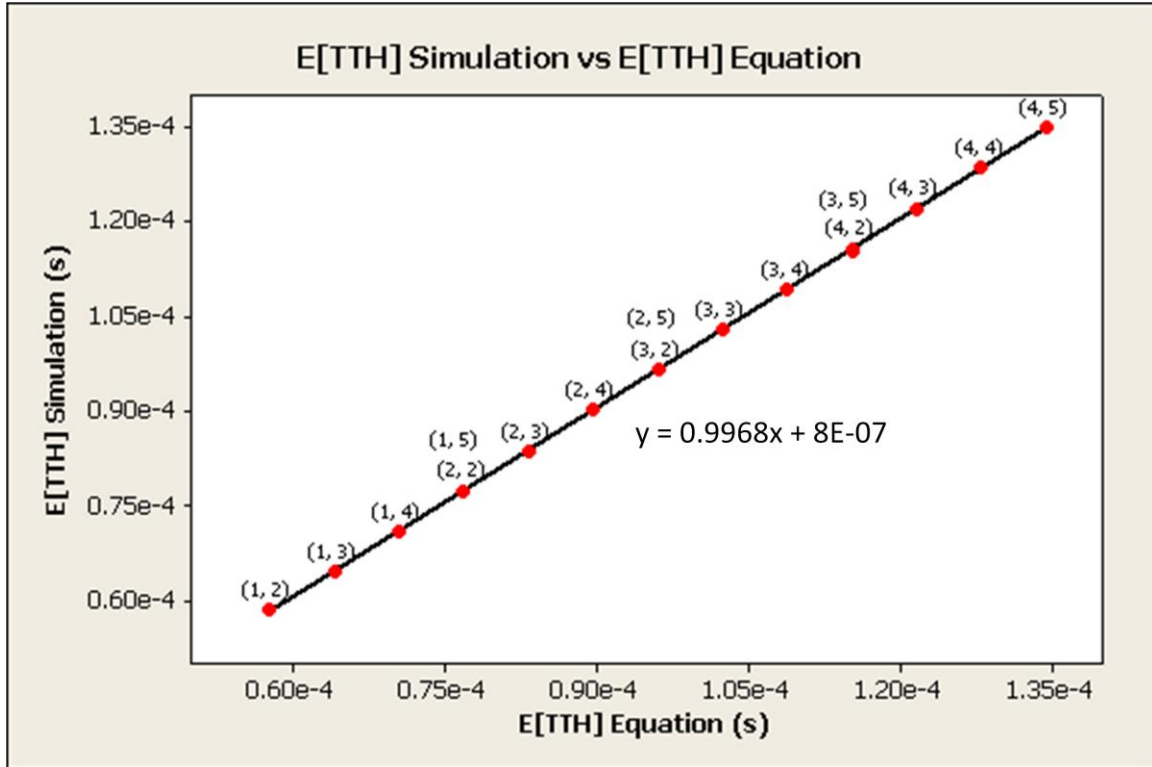
Figure 40: E[TTH] Equation vs. E[TTH] Simulation for various time slots (n, m)

4.2.4 Effects of Propagation Delay on E[TTR]

Section 4.1.3 detailed some of the benefits propagation delay can bring to the handshake. Figure 41 shows those effects. In this instance we used $n = 4, m = 4$. Other combinations of time slots would be affected differently by the same amount of propagation delay, but would mostly find an optimum point at 12.8 μs = τ. The

exception would be very low transmit and receive slots such as 2, in which the 12.8 μs

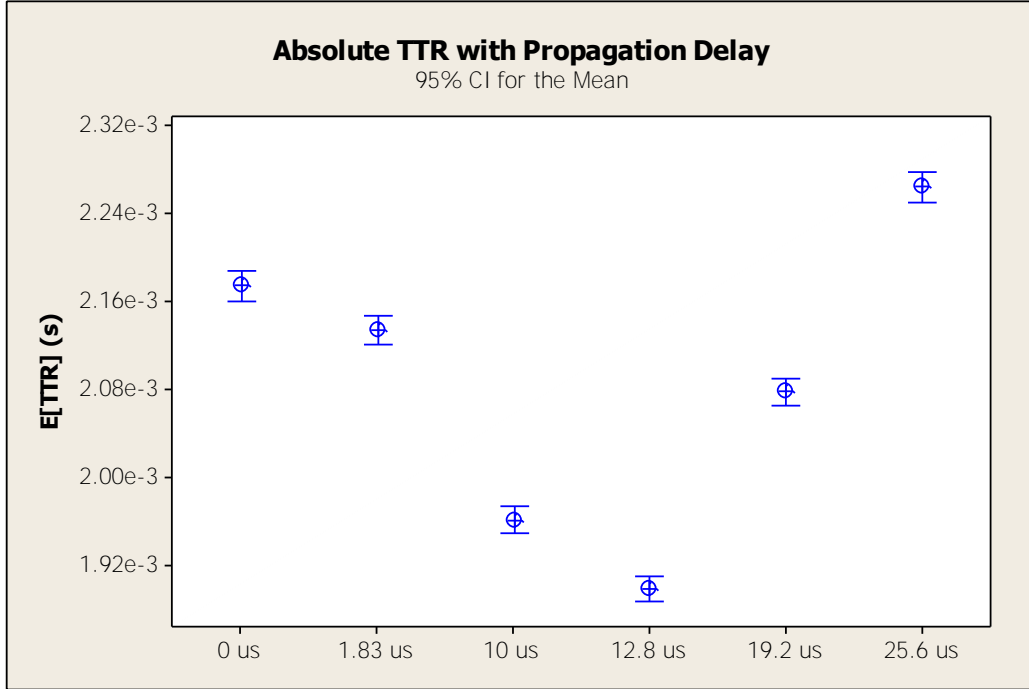propagation delay would cause the response to arrive after a radio left the channel.

Figure 41: TTR with n=4, m=4 with various amounts of propagation delay

4.2.5 Effects of Various Backoff Times

In Section 4.1.6 we discussed the effect of a uniform distribution of backoff times

should the radio timeout. As the amount of backoff time increases, the probability of the

handshake failing again decreases; however, spending too much backoff time in a

channel may increase the chances of either a rendezvousing radio or jammer discovering

the timed out radio. Figure 42 shows various amounts of backoff delay (where the

notation $n\tau = n$ time slots) with a radio using 4 transmit and receive slots. The minimal

backoff time for a single radio is 2τ (section 4.1.6), and the values closer to this result in a

higher TTR, because the probability of the handshake failing again is higher. Eventually,

the backoff time reaches an optimal point (~5τ and 7τ in this case) and slowly tapers off

76

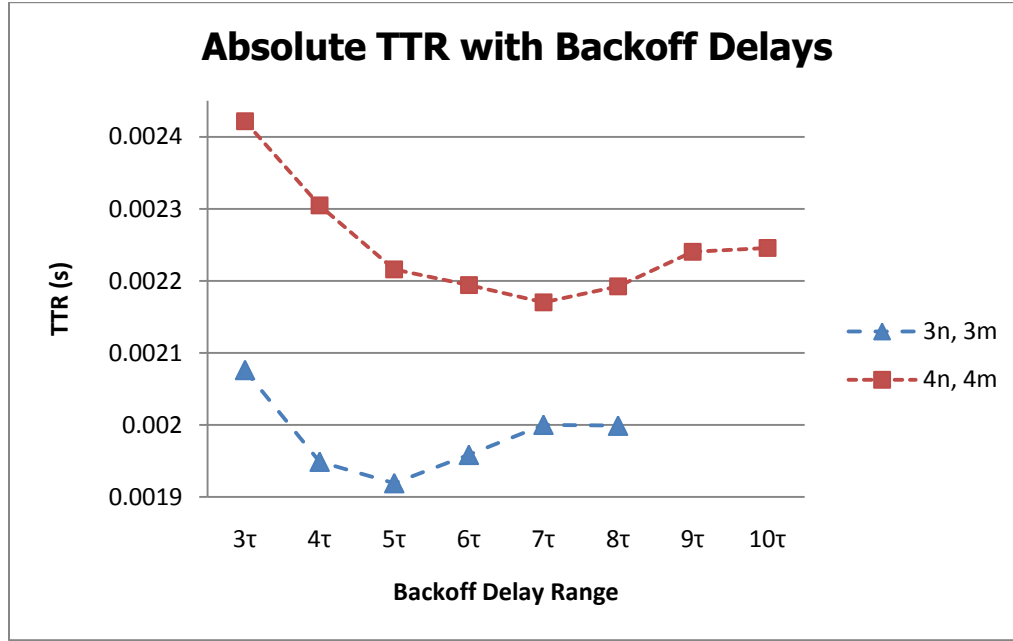afterwards. Various time slot configurations will have different optimal values for the backoff delay.

Figure 42: TTR using various backoff delays for the timeout; two different time slot configurations

4.3 Simulation Analysis of Jammer Effectiveness

Next we will analyze the effects of the jammers on the rendezvous and handshaking process. First we will discuss the expected effects of the jammers and then compare them with the actual results. Figure 43 and Figure 44 show the performance of each jammer. Jammer performance is compared based on the probability of jamming, the TTJ, and the increase in TTR over non-jammed TTR.

4.3.1 Effects of the Different Jammers

The noise jammer is expected to perform the worst of the four jammers. This is because the only time the noise jammer is effective is when it is in the same channel as both rendezvousing radios. The probability of this occurring is less than that of being in

the same channel as only one radio. However, the effect of the noise jammer is to completely disrupt rendezvous, forcing the radios to start over, which should cause a larger increase in TTR when the jammer is effective.

The deceptive jammer is likely to perform better than the noise jammer, however not as well as the sense jammer. The deceptive jammer works when in the same channel as any one radio, giving it a higher probability of jamming the rendezvousing radios; however, since the deceptive jammer only jams one radio at a time, the best it can do is to delay rendezvous. So while the deceptive jammer is more likely to jam than the noise jammer, the effects of each jam will not increase the TTR as much.

Being a combination of the first two, the sense jammer should perform better than both of them. This jammer has the probability of both delaying rendezvous and causing it to start over, each with the respective increases in TTR. Also, since this jammer will not be transmitting constantly like the first two, it will likely use less power to perform its job.

The PUE jammer is different in that it can have both negative and positive effects on the TTR. When the PUE jammer jams a single radio, that radio will remove whatever channel it was in from its list. This would cause the rendezvousing radios to have a differing set of available channels. Theis' showed in his research with the MC algorithm that having a different set of available channels increases TTR. This may also mean that certain rendezvous algorithms (i.e., Modified MC Algorithm [25]) would perform better than others in the presence of this particular jammer (particularly if the emulation is not carefully targeted). When the PUE jammer jams both radios at once, both radios would then remove the current channel from the list. Since both of them are removing the same

channel, this would reduce the effective search space for the rendezvous, thus decreasing TTR.
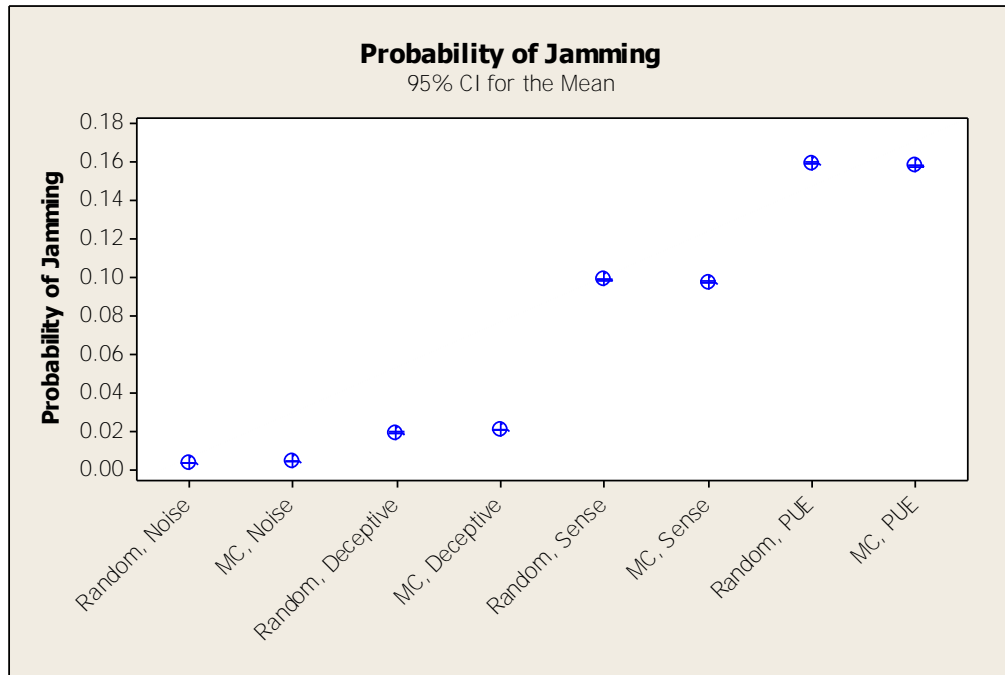

Figure 43: Probability of Jamming for the different jammers against random and MC rendezvous

As expected, the noise jammer performed the worst given that it had to have both radios in the same channel to be effective. The deceptive jammer performed slightly better because it could have any one radio in the same channel to jam. The sense jammer performed better than both of these given that it could be in the same channel as any or both radios to jam them. The PUE jammer performed better than expected. Since the PUE and sense jammers are both effective in the same instances (same channel as one radio or both) we would expect them to have similar probabilities of jamming. Also, our PUE jammer does not remove channels from its own list, meaning it will visit channels that the rendezvousing radios may have removed, meaning it should have an even lower probability.
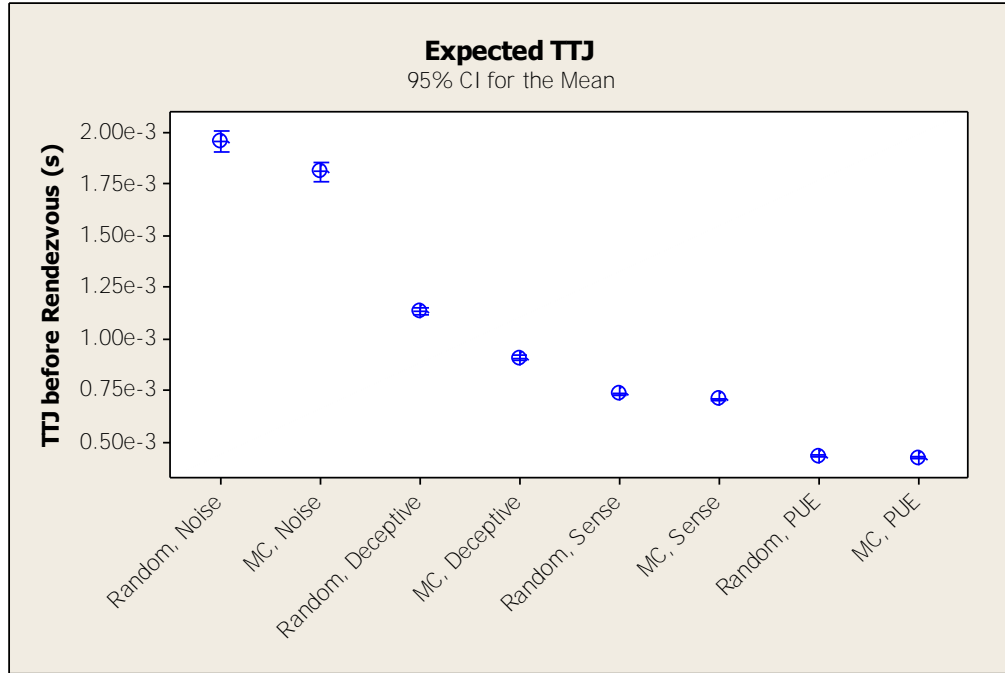
Figure 44: E[TTJ] before rendezvous for Random and MC Algorithms

Figure 44 is the opposite of the Figure 43. The jammers that have a lower probability of jamming will obviously have longer periods of time between jams before rendezvous has occurred.

These jammers are also not necessarily operating under optimal conditions. As discussed in Chapter 3, each jammer spends the same time in each channel as the rendezvousing radios. However, some jammers, such as the noise jammer, may not need to transmit very much noise in order to effectively disrupt the rendezvous process. This would allow it to spend less time in each channel and thus change channels much quicker compared to the rendezvousing radios, which in turn would likely increase its probability of jamming.

4.3.2 Effects on Different Rendezvous Algorithms

Initially it was thought that the various jammers would not affect different rendezvous algorithms differently. However, looking at Figure 45, we noticed that the jammers had a slightly greater affect on the MC algorithm than the random algorithm. We believe this is because for the Random Algorithm, each channel visited is independent of the previous channel. However, for the MC Algorithm, each subsequent channel is dependent upon the previous one as well as the rate and primes used. Therefore, when the jammer disrupted the radios using the MC Algorithm, it took longer for the rendezvousing radios to recover, as opposed to the Random Algorithm.

However, despite the greater effect of the jammers on the MC Algorithm, it still performs better than the Random Algorithm. Figure 46 shows the TTR for both Random and MC Algorithms, both without a jammer and with each of the jammers present.
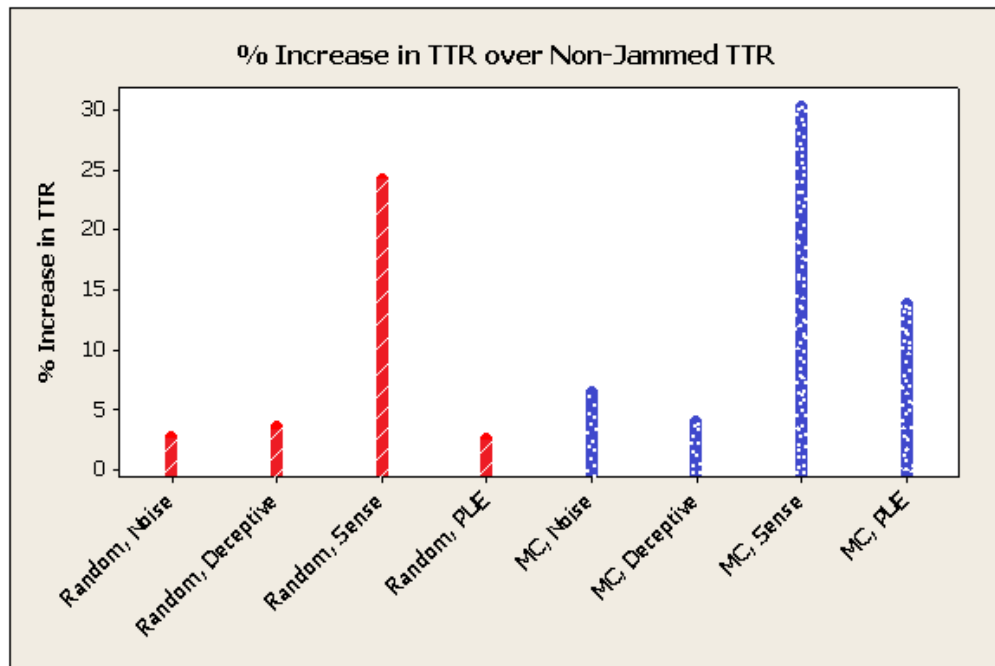

Figure 45: Percentage increase in TTR over non-jammed TTR due to Jammers for Random and MC Algorithms
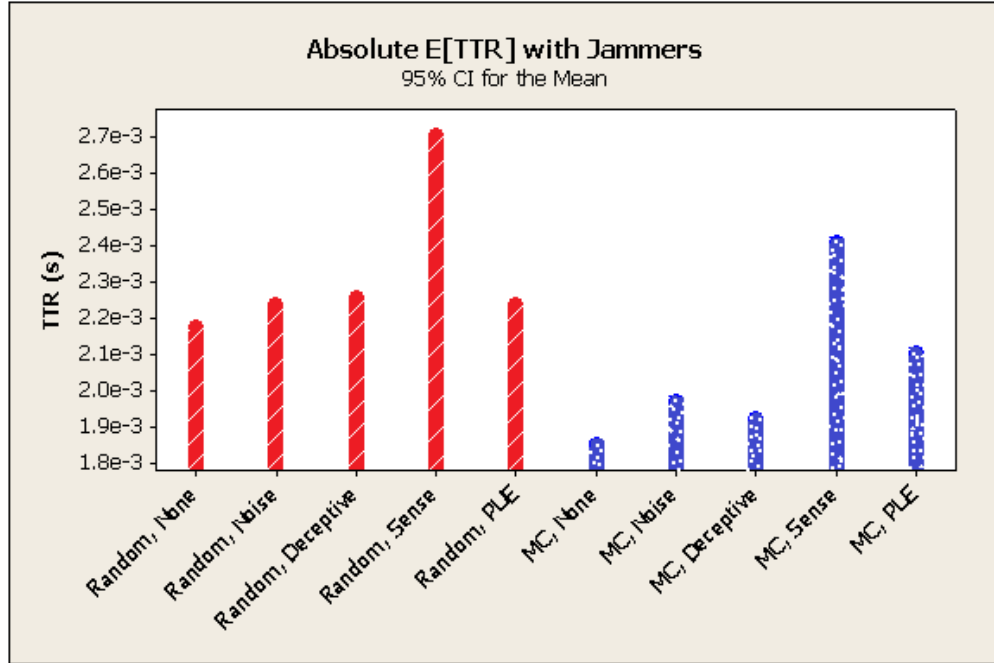
Figure 46: TTR for Random and MC Algorithms with Jammers

4.3.3 Jammer Comparison for Various Time Slots

Because of the effects the various time slots had on the number of steps need to rendezvous and TTR, it was believed that those setups that greatly increased the number of steps necessary would be more affected by a jammer due to the fact that the jammer would have more opportunities to jam the radios. Figure 47 shows the percentage increase in TTR due to the jammers for various time slot setups using random rendezvous. It was expected that the time slot setups with only two receive slots (those that caused a great increase in the number of steps) would have a larger percentage increase in TTR. However, this did not have the expected effect. The difference in number of steps for the various time slot setups likely did not overcome the fact that those setups that took longer also kept the rendezvousing radios in the same channel for longer, which can increase the chance for the jammer to find the rendezvousing radios.
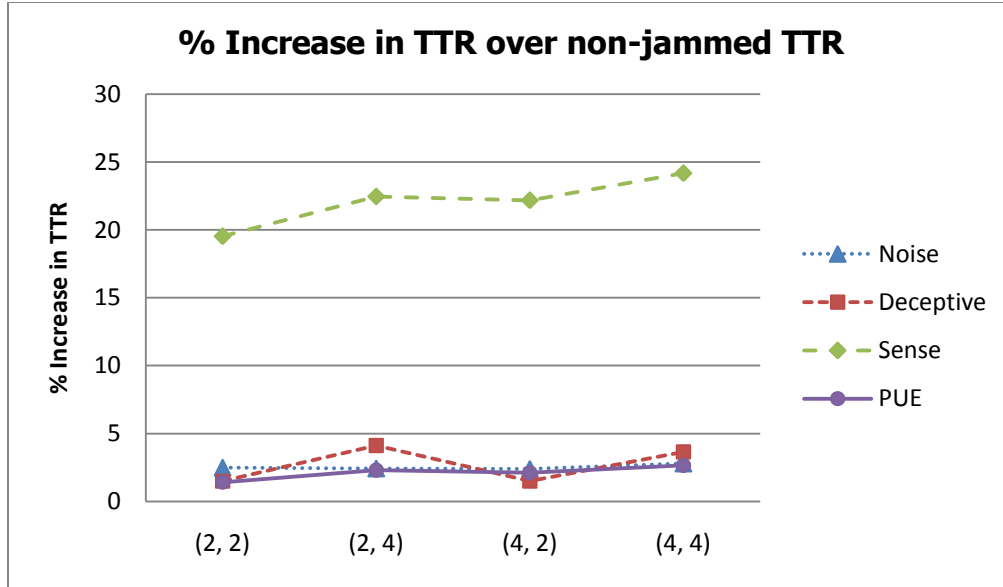
82

Figure 47: Percentage increase in TTR due to Jammers for various time slots (n, m) with Random Rendezvous

4.3.4 Jammer Effects with Different Number of Total Channels

The effect of an increase number of channels that rendezvousing radios can meet in is well known as it increases the possible search space for the two radios. The effect is likely to be the same for the jammers, making it less likely to find the rendezvousing radios for larger number of channels. However, the extent to which the increase in number of channels affects each jammer is interesting. In Figure 48, as the number of channels increase, the effect of the first three jammers decreases rapidly as expected. However, the PUE jammer seems to increase in effectiveness as the number of channels increase. This is likely because the possibility of reducing the search space has a greater impact at lower numbers of channels where the search space is already small.
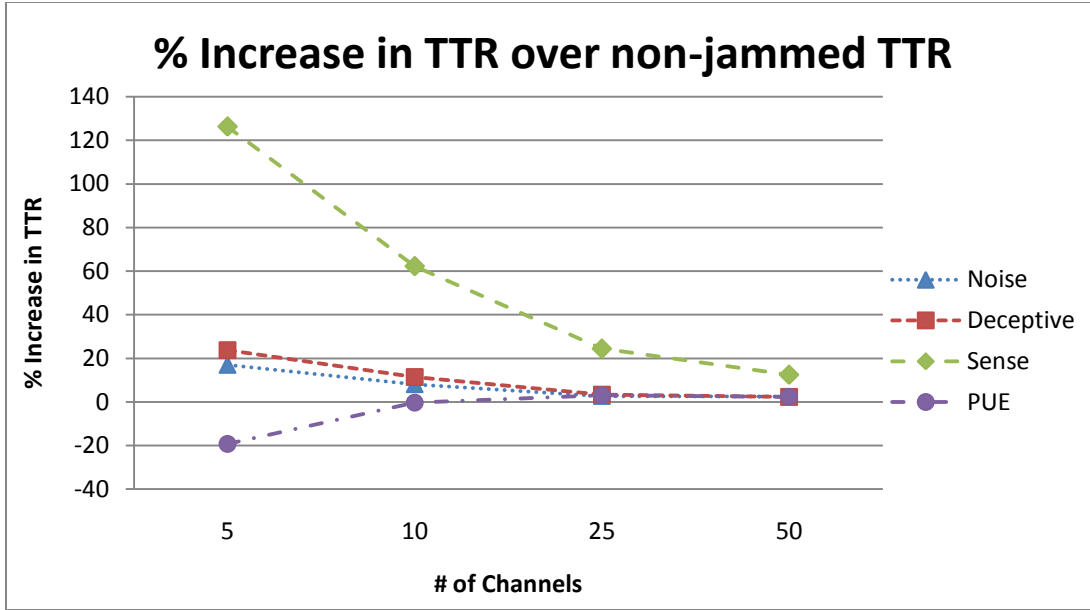
Figure 48: Percentage increase in TTR over non-jammed TTR from each jammer for different number of channels

4.3.5 Expected Duty Cycle for Sense Jammer

As discussed in Section 3.3.1, the E[DC] is a function of the probability of receiving particular beacons, and the time spent responding to those beacons. Figure 49 shows the E[DC] for the sense jammer during simulation. In this case, the sense jammer only transmits a single response beacon if it received an initial beacon, and only transmits noise for one time slot (τ) when it received a response beacon. More transmission time may be needed, in which case the E[DC] would be higher. Even with an increase, the E[DC] is extremely low compared to the 100% of the other jammers. While power will still be used during its sensing phase, it will likely not use nearly as much as when transmitting.
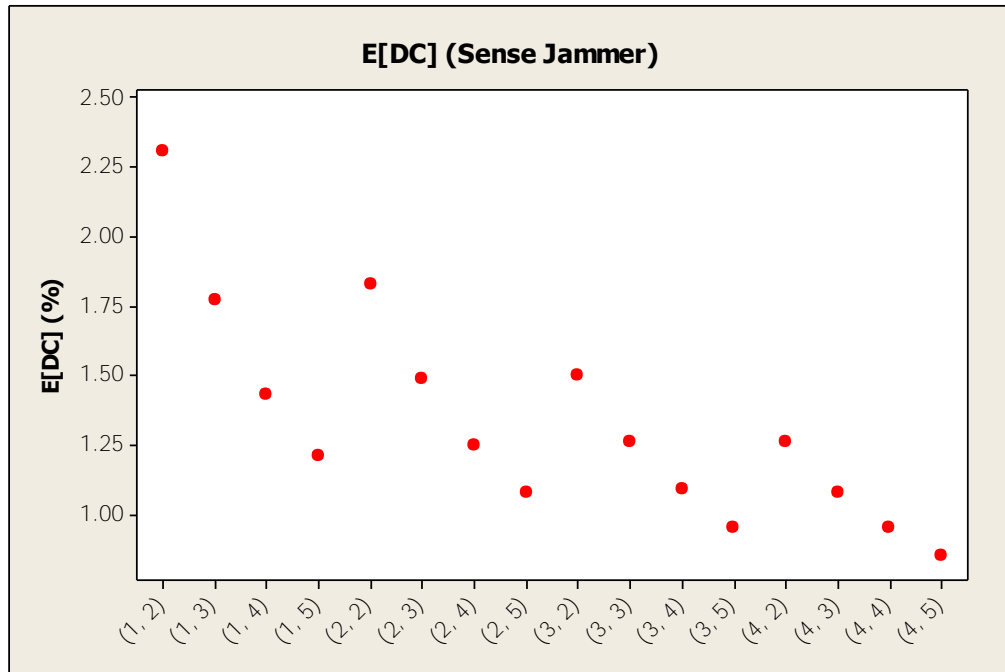
Figure 49: E[DC] for sense jammer with various time slots (n, m), 25 channels, assumed one time slot (τ) of jamming transmission time each time it attempts to jam

V. Conclusions

This chapter reviews the research topic and goals exhibited in this thesis. It reiterates the handshake and jamming algorithm development, as well as the results obtained from mathematical analysis and simulation.

5.1 Research Goals

The goal of this thesis was to continue the development of blind rendezvous protocols for DSA, formalizing the handshake necessary to complete rendezvous and investigating its performance in a hostile environment. This goal was achieved by developing and characterizing a handshaking algorithm that two radios can use to setup communications when in the same channel. Furthermore, we investigated the handshake and rendezvous performance under a series of jamming techniques.

5.2 Research Results

We have successfully presented an effective handshaking technique to be used by two radios attempting blind rendezvous. The handshake completes the rendezvous process, providing the final step to allowing two radios to find one another in a DSA environment.

We began our analysis with a mathematical analysis of the handshaking algorithm to investigate this possibility by determining the probability of a successful handshake and the expected TTH. We found that as the number of transmit and receive slots in the handshake increased, the probability of a successful handshake also increased; however, it also took longer to complete the handshake. These analytical results were validated by a more detailed simulation that produced similar results.

The simulation also revealed additional insight: it showed that as the number of time slots for transmitting and receiving increased, the number of steps (channel changes)

necessary to rendezvous decreased. However, the total time (as opposed to steps) to rendezvous still increased because each radio was spending more time in each channel due to the additional slots. Overall, the handshake took between 40 to 360% longer to rendezvous (as opposed to meet) depending on the number of transmit and receive slots used.

The handshake process was also tested under varying propagation delays and backoff delays. Propagation delay had positive and negative effects on the handshake. With the exception of low numbers of receive slots ($m = 2$), up to a certain point ($d = \tau$), propagation delay would shrink the handshake failure window, thus increasing the probability of a handshake. After this, propagation delay begins to cause overlap between the receive windows of the radios, decreasing the probability of a handshake, until eventually radios would change channels before beacons and responses would arrive.

In order to decorrelate attempts to handshake (so that two radios did not get stuck in an offset inside the handshake failure window), a backoff delay was implemented to change the time offsets between the two rendezvousing radios. The backoff used was a uniform distribution of delays. The optimal amount of backoff time was dependent on the number of transmit and receive slots being used.

Four different jamming algorithms were developed to use against the handshake. The primary metric used was the increase in TTR due to each jammer over non-jammed TTR. We also examined the probability of jamming and the expected TTJ for each jammer. As expected, each jammer increased the TTR. The amount of increase was greatly affected by the total number of channels available. The noise, deceptive, and

sense jammers were more effective with fewer channels, whereas the PUE jammer increased in effectiveness as the channels increased. During random rendezvous, the noise, deceptive, and PUE jammers were the least effective, increasing TTR at most by 4%. The sense jammer performed the best, increasing TTR by about 25%. When used against the MC algorithm, the jammers were even more effective. The noise and deceptive jammers still performed the poorest, but now increased TTR up to 7%. The sense jammer again performed the best with a 30% increase. The PUE jammer was in the middle with a 14% increase, nearly 10% more than when used against the random algorithm.

5.3 Research Significance

Spectrum scarcity and DSA have increased the need for efficient means for cognitive systems to rendezvous. This research advances solutions to the blind rendezvous problem that are used to solve these problems. It provided a unique look at the rendezvous process that allows for more accurate analysis of rendezvous capabilities that will hopefully lead to more efficient rendezvous methods. Cognitive systems may not always operate in friendly environments either. Radio interference and jamming can pose a serious threat to communications. The jamming techniques used in this thesis are simple, yet have a significant impact on the rendezvous process. This indicated that proper defense of rendezvousing systems is necessary.

5.4 Future Research

Three areas of future research are further refinement of the handshake and jamming algorithms and hardware implementation. To refine the handshake, the backoff delay ranges should be examined more closely. Our backoff delays used a uniform

distribution, but other distributions may prove more effective, and specific relationships between the numbers of transmit and receive slots and the optimal backoff delays could be discovered.

To refine the jammers, more realistic algorithms/implementations should be made. Our jammer functionality was assumed; we did not analyze specific message formats or noise transmission. The effects of the jammers are likely to change given a more realistic scenario. In the same vein, our analysis of expected duty cycle was also simple, and was only applied to the sense jammer. A more robust analysis could include looking at the power consumption of each jammer. Depending on the type of messages, or the amount of power necessary to function, each jammer may have better performance to power consumption ratios. Also, for simplicity, each jammer spent the same amount of time in each channel as the rendezvousing radios. Analyzing the effects of jammers spending various amounts of time in the channel compared to the rendezvousing radios would be helpful.

Hardware implementation would allow for better real world analysis of the effects of both the handshake protocol and jamming techniques used against it, to include the power consumption analysis mentioned. Hardware implementation may also allow for more sophisticated jamming techniques that were not practical in simulation. Hardware implementations would also allow for better investigation into defending against these jamming techniques as many occur in the physical layer.

References

[1] Akyildiz, Ian F., Won-Yeol Lee, Mehmet C. Vuran, and Shantidev Mohanty. "NeXt generation/dynamic spectrum access/cognitive radio wireless networks: A survey." *Computer Networks 50*, 2006: 2127-2159.

[2] Alonso, G., E. Kranakis, R. Wattenhofer, and P. Widmayer. "Probabilistic Protocols for Node Discovery in Ad-hoc, Single Broadcast Channel Networks (Extended Abstract)." *IEEE Computer Society*, 2003.

[3] Alpern, Steve, and Wei Shi Lim. "The Symmetric Rendezvous-Evasion Game." *SIAM J. Control Optim.*, 1998: 948-959.

[4] Anderson, E. J., and R. R. Weber. "The Rendezvous Problem on Discrete Locations." *Journal of Applied Probability*, 1990: 839-851.

[5] Bellardo, John, and Stefan Savage. "802.11 Denial-of-Service Attacks: Real Vulnerabilities and Practical Solutions." Washington, D.C.: The USENIX Association, 2003.

[6] Borbash, Steven A., Anthony Ephremides, and Michael J. McGlynn. "An asynchronous neighbor discovery algorithm for wireless sensor networks." Elsevier B.V. Vol. 5, no. 7, 2006.

[7] Buddhikot, Milind M., Paul Kolodzy, Scott Miller, Kevin Ryan, and Jason Evans. "DIMSUMNet: New Directions in Wireless Networking Using Coordinated Dynamic Spectrum Access." Proceedings of the 6th IEEE International Symposium on a World of Wireless Mobil and Multimedia Networks, 2005.

[8] "Cognitive Jammer Fundamental Research Results." Echo Ridge, 2009.

[9] Dahm, Werner J.A., and Eliahu Niewood. *Implications of Spectrum Management for the Air Force.* United States Scientific Advisory Board, 2008.

[10] DaSilva, L. A., and I. Guerreiro. "Sequence-based Rendezvous for Dynamic Spectrum Access." *Third IEEE DYSPAN*, 2008: 1-7.

[11] DaSilva, Luiz A., and Ryan W. Thomas. "Rendezvous in Cognitive Radio Networks." In *Cognitive Radio Technology*, by Bruce A. Fette, 635-644. Amsterdam: Elsevier, 2009.

[12] Fette, Bruce A. "History and Background of Cognitive Radio Technology." In *Cognitive Radio Technology*, by Bruce A. Fette, 1-23. Amsterdam: Elsevier, 2009.

[13] Gal, S., and J. V. Howard. "Rendezvous-Evasion Search in Two Boxes." *Operations Research*, 2005: 689-697.

[14] Gast, Matthew S. *802.11 Wireless Networks: The Definitive Guide.* Beijing: O'Reilly Media Inc, 2005.

[15] Goswami, Subrata. *Internet Protocols: Advances, Technologies, and Applications.* Boston: Kluwer Academic Publishers, 2003.

[16] Jing, Xiangpeng, Siun-Chuon Mau, and Robert Matyas. "Reactive Cognitive Radio Algorithms for Co-Existence between IEEE 802.11b and 802.16a Networks." *IEEE GLOBECOM*, 2005.

[17] Liyana Arachchige, Chanaka J., S. Venkatesan, and Neeraj Mittal. "An Asynchronous Neighbor Discovery Algorithm for Cognitive Radio Networks." *3rd IEEE Symposium on New Frontiers in Dyanmic Spectrums Access Networks (DySPAN).* Chicago, 2008. 1-5.

[18] Ma, Liangping, and Chien-Chung Shen. "Security-enhanced Virtual Channel Rendezvous Algorithm for Dynamic Spectrum Access Wireless Networks." *3rd IEEE Symposium on New Frontiers in Dynamic Spectrum Access Networks (DySPAN).* San Diego, 2008. 1-9.

[19] Mitola III, Joseph. "Cognitive Radio for Flexible Mobile Multimedia Communications." Springer Netherlands. Vol 6, no. 5, 2001.

[20] Sampath, Ashwin, Hui Dai, Haitao Zheng, and Ben Y. Zhao. "Multi-channel Jamming Attacks using Cognitive Radios." Proceedings of 16th International Conference on Computer Communications and Networks, 2007, pp. 352-357.

[21] Silvius, Mark D., Allen B. MacKenzie, and Charles W. Bostian. "Rendezvous MAC Protocols for Use in Cognitive Radio Networks." IEEE Military Communications Conference, 2009.

[22] Stark, Henry, and John W. Woods. *Probability, Random Processes, and Estimation Theory for Engineers 2nd Ed.* New Jersey: Prentice-Hall, Inc, 1994.

[23] Stevens, W. Richard. *TCP/IP Illustrated, Volume 1.* Boston: Addison Wesley, 1994.

[24] Swartz, Mischa. *Mobile Wireless Communications.* Cambridge: Cambridge University Press, 2005.

[25] Theis, Nicholas C. "The Modular Clock Algorithm for Blind Rendezvous." Wright-Patterson AFB, Ohio: Air Force Institute of Technology, 2009.

[26] Theis, Nick C., Ryan W. Thomas, and Luiz A. DaSilva. "Rendezvous for Cognitive Radios." *Transactions on Mobile Computing*, 2009: 1-19.

[27] Thomas, Ryan W, Daniel H. Friend, Luiz A. DaSilva, and Allen B. MacKenzie. "Cognitive Networks: Adaptation and Learning to Achieve End-to-End Performance Objectives." *IEEE Communications Magazine*, 2006: 1-9.

[28] Weber, Richard. "The optimal strategy for symmetric rendezvous search on K3." 2007.

[29] Xu, Wenuyuan, Wade Trappe, Yanyong Zhang, and Timothy Wood. "The Feasibility of Launching and Detecting Jamming Attacks in Wireless Networks." Proceedings of the 6th ACM International Symposium on Mobile Ad-hoc Networking and Computing, 2005, pp. 46-57.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 074-0188

1. REPORT DATE (DD-MM-YYYY) 25-03-2010	2. REPORT TYPE **Master's Thesis**	3. DATES COVERED (From – To) Aug 2008 – Mar 2010

4. TITLE AND SUBTITLE Handshaking Protocols and Jamming Mechanisms for Blind Rendezvous in a Dynamic Spectrum Access Environment	5a. CONTRACT NUMBER
	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S) Gross, Aaron A., 2d Lt, USAF	5d. PROJECT NUMBER 10ENG110
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAMES(S) AND ADDRESS(S) Air Force Institute of Technology Graduate School of Engineering and Management (AFIT/EN) 2950 Hobson Way WPAFB OH 45433-7765	8. PERFORMING ORGANIZATION REPORT NUMBER AFIT/GCO/ENG/10-09

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) Dr. Vasu Chakravarthy Air Force Research Laboratory 2241 Avionics Circle, Bldg 620 WPAFB, OH 45433-7318 937-255-2620 x4067; Vasu.Chakravarthy@wpafb.af.mil	10. SPONSOR/MONITOR'S ACRONYM(S) AFRL/RYRE
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT **Approved for Public Release; Distribution Unlimited.**

13. SUPPLEMENTARY NOTES

14. ABSTRACT
Blind frequency rendezvous is an important process for bootstrapping communications between radios without the use of pre-existing infrastructure or common control channel in a Dynamic Spectrum Access (DSA) environment. In this process, radios attempt to arrive in the same frequency channel and recognize each other's presence in changing, under-utilized spectrum. This paper refines existing blind rendezvous techniques by introducing a handshaking algorithm for setting up communications once two radios have arrived in the same frequency channel. It then investigates the effect of different jamming techniques on blind rendezvous algorithms that utilize this handshake. The handshake performance is measured by determining the probability of a handshake, the time to handshake, and the increase in time to rendezvous (TTR) with a handshake compared to that without. The handshake caused varying increases in TTR depending on the time spent in each channel. Four different jamming techniques are applied to the blind rendezvous process: noise, deceptive, sense, and Primary User Emulation (PUE). Each jammer type is analyzed to determine how they increase the TTR, how often they successfully jam over a period of time, and how long it takes to jam. The sense jammer was most effective, followed by PUE, deceptive, and noise, respectively.

15. SUBJECT TERMS **Handshaking, Blind Rendezvous, Dynamic Spectrum Access, Jamming**

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON **Ryan W. Thomas, Maj, USAF (ENG)**
REPORT U	ABSTRACT U	c. THIS PAGE U	UU	108	19b. TELEPHONE NUMBER (Include area code) 937-255-3636 x4613; e-mail: Ryan.Thomas@afit.edu

Standard Form 298 (Rev: 8-98)

Prescribed by ANSI Std. Z39-18